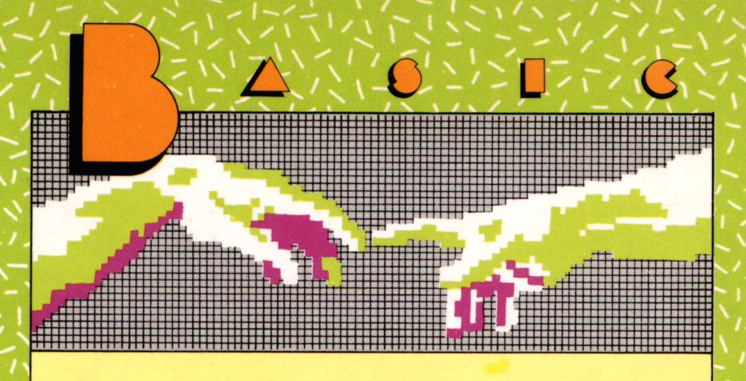


GRÁFICOS COLORES Y MÚSICA en el ZX Spectrum

E. Lowy Frutos — A.E. Gallego Palomero
S. Mansilla Romo

LeG



750



CAJA DE AHORROS Y
MONTE DE PIEDAD DE MADRID

BIBLIOTECA CENTRAL

Signatura

Registro 14.211

Autor: LOWY FRUTOS, E.

Título: Gráficos, colores y música en el ZX Spectrum.

LECTOR		FECHA	
Sala	Préstamo	Entregado	Devuelto

T... Gráficos Colores y musica.

Autor Lowy Frutos.

Editorial S.M.

Distribuidor 16K5

Precio

Adjuntar este vale al pedido

Clave

Cant. ejem.

EQUIPO DE AUTORES

Ernesto Lowy Frutos
Agregado de Física y Química de I.B.

A. Enrique Gallego Palomero
Agregado de Matemáticas de I.B.

Serafín Mansilla Romo
Licenciado en Matemáticas

EQUIPO EDITORIAL

Coordinación: José Luis Robles Cid

Maqueta: José Ugarte

Dibujos: Ernesto Cobeño

Fotografías: José Manuel Navia, AISA

Portada: Alfonso Ruano y José Luis Cortés



GRÁFICOS COLORES Y MÚSICA en el ZX Spectrum

E. Lowy Frutos — A.E. Gallego Palomero
S. Mansilla Romo

I.S.B.N.: 84-348-1622-9 / Depósito legal: M. 20.389-1985 / Fotocomposición: Grafilia, S. L.
Imprime: Gráficas VELASCO, S. A. - Antonio de Cabezón, 13 - 28034 Madrid.
Impreso en España - Printed in Spain.

PRESENTACIÓN

Una función importante que puede cumplir un microordenador es la de dibujar en la pantalla distintos tipos de gráficos.

El control de esta posibilidad ofrece al usuario la ocasión de afianzar su práctica en la programación y la de estimular su imaginación y creatividad en la exploración de la capacidad del microordenador en la realización de dibujos. Además, esta máquina es capaz de generar una amplia gama de colores en la pantalla y de producir sonidos y música.

La utilización simultánea de todas estas posibilidades en un mismo programa conduce a una creativa actividad de análisis y síntesis, a despertar facultades estéticas y a disfrutar del ejercicio de las mismas. Aunque inicialmente todos estos procesos se ponen en marcha en un contexto técnico, su posterior desarrollo supone el fomento de aspectos educativos de indudable valor formativo, a cuyo afianzamiento desea contribuir este libro.

En mayor o menor medida, casi todos los microordenadores poseen la capacidad de generar GRÁFICOS, COLORES y MÚSICA. Sin embargo, el modo y la forma de presentarlos son específicos de cada microordenador; y dado que en el momento actual el más difundido es el ZX *Spectrum,* los programas incluidos en este libro están preparados para funcionar en este microordenador. No obstante, es factible adaptarlos a otros microordenadores, para lo cual se sugiere consultar el manual correspondiente.

Se ha tenido especial cuidado en dotar a este libro de unas características afines a nuestro entorno cultural. Esto se ha intentado plasmar tanto en aspectos generales como en detalles concretos. Entre los primeros cabe destacar el estilo del lenguaje, en el que se ha evitado incluir giros y vocablos foráneos innecesarios, y entre los segundos, la elección de piezas musicales de nuestro folclore.

En este libro, como en el anterior de la misma serie, se ha seguido el método inductivo, introduciendo las instrucciones de GRÁFICOS, COLORES y MÚSICA a partir de programas sencillos, todos ellos resueltos. También se ha procurado dar a esta obra un enfoque interdisciplinar, mediante el desarrollo de programas relacionados con otras áreas de conocimiento.

LOS AUTORES

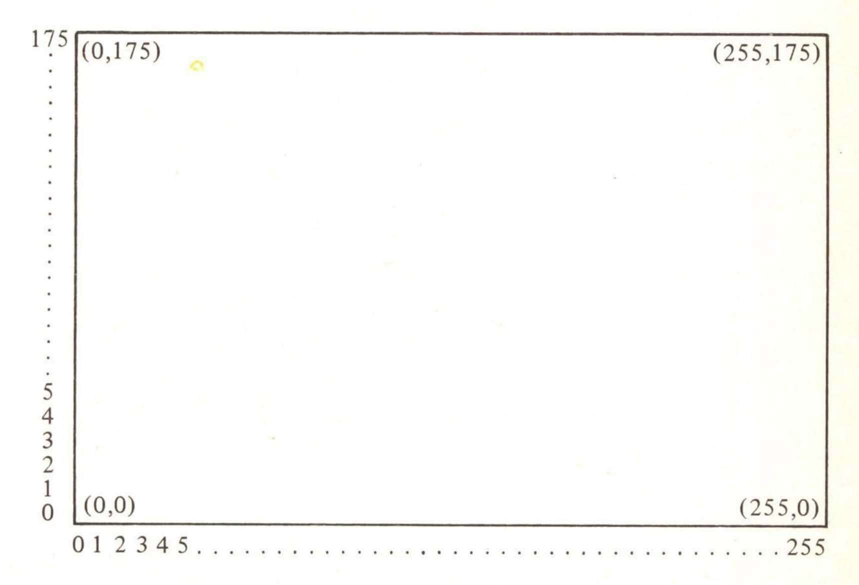
1. Gráficos en alta resolución

1. Pantalla de alta resolución

- En ***baja resolución,*** el ZX Spectrum escribe en 22 filas y 32 columnas, lo cual permite localizar 22 × 32 = 704 elementos en la pantalla: asteriscos, puntos u otros caracteres. Esta cantidad de elementos disponibles en baja resolución es escasa a la hora de dibujar gráficos, pues éstos no alcanzan suficiente precisión.

  No obstante, con ciertas instrucciones el microordenador puede imprimir puntos en 176 líneas horizontales (de 0 a 175) y en 256 líneas verticales (de 0 a 255); luego en la pantalla se pueden localizar 176 × 256 = = 45056 puntos. En estas circunstancias se dice que el microordenador imprime en ***alta resolución,*** y los gráficos que dibuja, ***gráficos en alta resolución.***

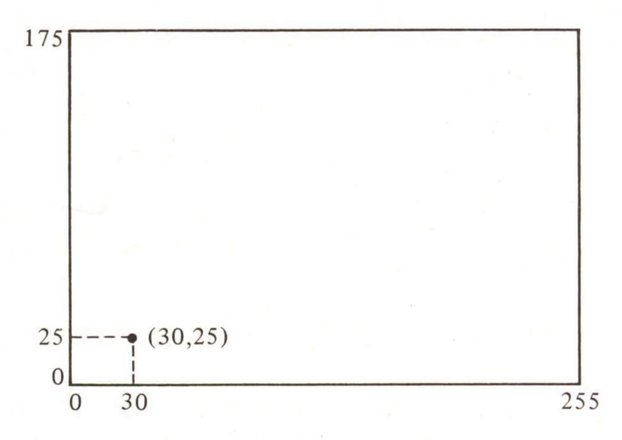
  La pantalla de alta resolución del microordenador ZX Spectrum está referida a un sistema de coordenadas, cuyo origen está localizado en el ***vértice inferior izquierda*** (fila 0, columna 0); a los demás vértices les corresponden las coordenadas indicadas en la figura.



Las cuatro esquinas o vértices de la pantalla tienen, en consecuencia, las siguientes coordenadas:

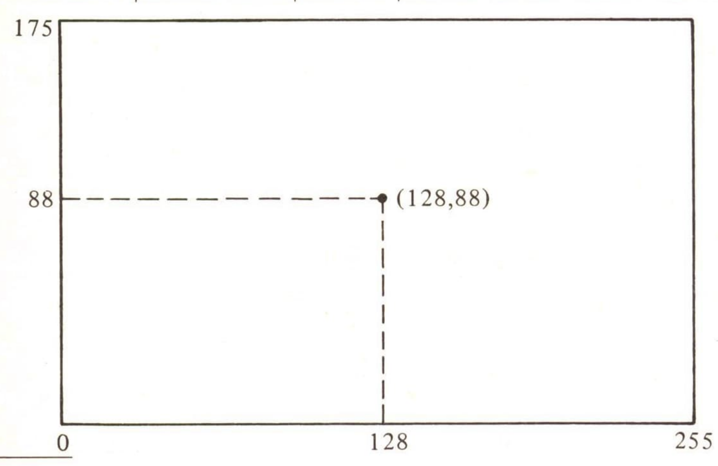
— Inferior izquierda: (0, 0).
— Inferior derecha: (255, 0).
— Superior izquierda: (0, 175).
— Superior derecha: (255, 175).

- A cada elemento de la pantalla, en alta resolución, se le llama ***pixel*** (*). Así, el pixel de coordenadas (30, 25) es el lugar de la pantalla separado del origen ***30 unidades en horizontal y 25 en vertical.***



Luego la primera coordenada de pixel es la ***abscisa*** y la segunda la ***ordenada.***

El centro de la pantalla corresponde al pixel de coordenadas (128, 88).



(*) Esta palabra resulta de la contracción de la expresión inglesa «picture cell», que viene a significar ***célula imagen.***

2. Instrucción para imprimir puntos

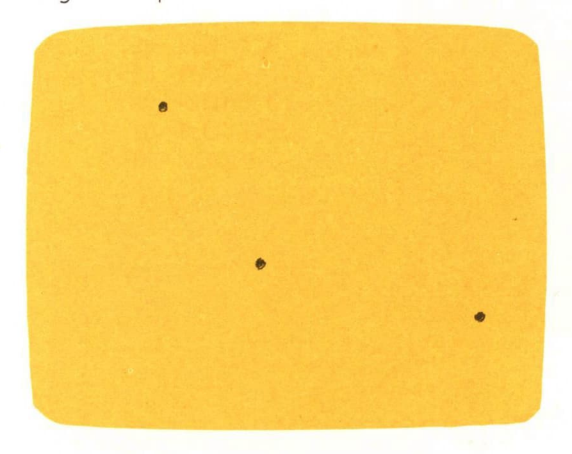
- El primer problema que surge en alta resolución es el de imprimir un punto en unas coordenadas determinadas. Esto se consigue con la instrucción

  PLOT X, Y

  que imprime un punto en el pixel «de horizontal X y de vertical Y».
- Aplicaremos esta instrucción en el siguiente programa que imprime puntos en las coordenadas que se van introduciendo.

```
10  INPUT "HORIZONTAL"; X
20  INPUT "VERTICAL"; Y
30  PLOT X, Y
40  GO TO 10
```

Si se ejecuta el programa (pulsar **RUN** y **ENTER**) y se introducen las coordenadas (200, 0), (128, 88) y (0, 100), en la pantalla se verán los siguientes puntos:



Debido a la instrucción 40 GO TO 10 este programa sigue pidiendo pares de valores X e Y e imprimiendo puntos.

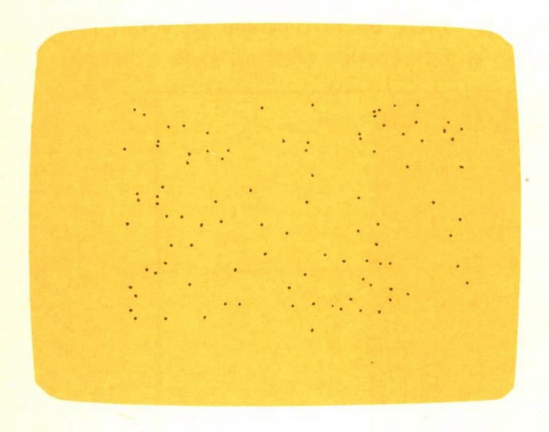
- Este otro programa imprime 100 puntos en la pantalla de forma aleatoria.

```
10 RANDOMIZE
20 FOR I = 1 TO 100
30 LET X = INT (RND * 256)
40 LET Y = INT (RND * 176)
50 PLOT X, Y
60 NEXT I
```

Si se ejecuta el programa (teclear **RUN** y **ENTER**) aparecerán en la pantalla 100 puntos, los cuales irán variando cada vez que se ejecute el programa, debido a la instrucción **10 RANDOMIZE** y a la función **RND** que aparecen en las instrucciones **30** y **40,** respectivamente.

Las coordenadas X, Y de cada punto se obtienen tomando la parte entera **INT** del producto de **RND** por el número de puntos del eje horizontal, 256 (de 0 a 255) y por el número de puntos del eje vertical, 176 (de 0 a 175), respectivamente.

Una de las múltiples pantallas que se pueden obtener al ejecutar el programa es ésta:

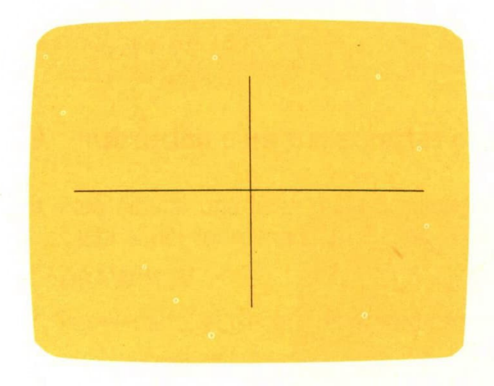


PROGRAMAS RESUELTOS

1. Teniendo en cuenta que el centro de la pantalla es el punto (128, 88), ¿qué dibuja el siguiente programa?

```
 1 REM EJES COORDENADOS CENTRADOS EN EL PUNTO 128,88
10 FOR X = 0 TO 255
20 PLOT X, 88
30 NEXT X
40 FOR Y = 0 TO 175
50 PLOT 128, Y
60 NEXT Y
```

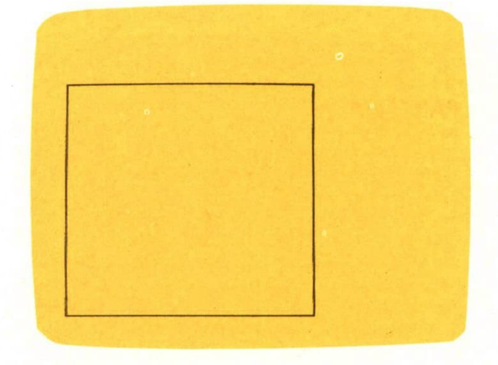
Solución



2. Hacer un programa que dibuje un cuadrado de vértices (0, 0), (175, 0), (0, 175) y (175, 175).

Solución

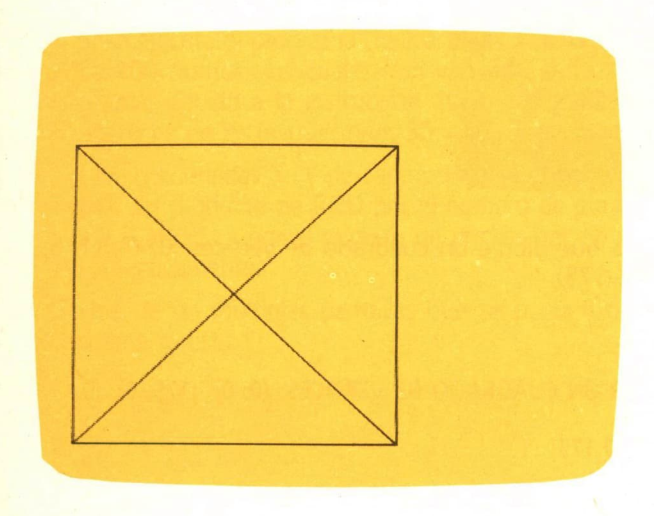
```
 1 REM DIBUJAR UN CUADRADO DE VERTICES: (0, 0), (175, 0), (0,
175) Y (175, 175)
10 FOR K = 0 TO 175
20 PLOT K, 0
30 PLOT K, 175
40 PLOT 0, K
50 PLOT 175, K
60 NEXT K
```



3. Añadir al cuadrado del ejercicio anterior sus diagonales.

Solución

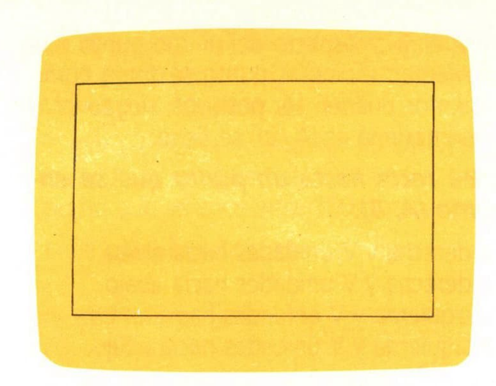
```
 1 REM DIBUJAR UN CUADRADO DE VERTICES: (0, 0), (175, 0), (0,
175) Y (175, 175); Y SUS DIAGONALES
10 FOR K = 0 TO 175
20 PLOT K, 0
30 PLOT K, 175
40 PLOT 0, K
50 PLOT 175, K
60 PLOT K, K
70 PLOT K, 175-K
80 NEXT K
```



4. Hacer un programa que recuadre la pantalla.

Solución

```
 1 REM RECUADRAR TODA LA PANTALLA
10 FOR X = 0 TO 255
20 PLOT X, 0
30 PLOT X, 175
40 NEXT X
50 FOR Y = 0 TO 175
60 PLOT 0, Y
70 PLOT 255, Y
80 NEXT Y
```



3. Instrucción para trazar rectas o segmentos de recta

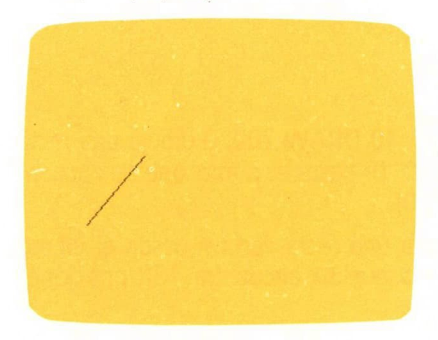
- Para dibujar una línea recta desde el último punto (A, B) que se imprimió hasta el del coordenada A + H, B + V se utiliza la instrucción:

  DRAW H, V

  Por ejemplo, al ejecutar el programa:

```
10 PLOT 20, 30
20 DRAW 40, 50
```

  se obtiene un segmento de recta de extremos (20, 30) y (20 + 40, 30 + 50) = (60, 80).



Vemos que la instrucción **DRAW** es relativa respecto del último punto que se imprimió (si no se imprimió previamente ningún punto se toma como punto de partida (0, 0)). Sus argumentos pueden ser positivos, negativos o nulos. Así, si el último punto que se imprimió es (A, B), se tiene:

	Dibuja una línea recta hasta un punto que se encuentra del punto (A, B):
DRAW H, V	H unidades a la derecha y V unidades hacia arriba.
DRAW H,-V	H unidades a la derecha y V unidades hacia abajo.
DRAW-H, V	H unidades a la izquierda y V unidades hacia arriba.
DRAW-H,-V	H unidades a la izquierda y V unidades hacia abajo.

Si en la instrucción **DRAW H, V,** H es 0, se imprime un segmento vertical, dado que la abscisa se mantiene constante, y es la que corresponde a la del punto anterior impreso. Por el contrario, si V es 0 se imprime una recta horizontal por permanecer la ordenada constante.

En la utilización de esta instrucción, como de otras que estudiaremos más adelante, hay que evitar que las coordenadas de los puntos adquieran valores que sitúen a éstos fuera de la pantalla, pues ello ocasionaría la interrupción de la ejecución, acompañada de un mensaje de error. Es necesario, por tanto, asignar a los argumentos H y V valores apropiados que eviten salirse de la pantalla.

- El siguiente programa muestra el funcionamiento de la instrucción DRAW con distintos argumentos.

```
 1 REM RECTANGULO
10 DRAW 200, 0
20 DRAW 0, 100
30 DRAW −200, 0
40 DRAW 0, −100
```

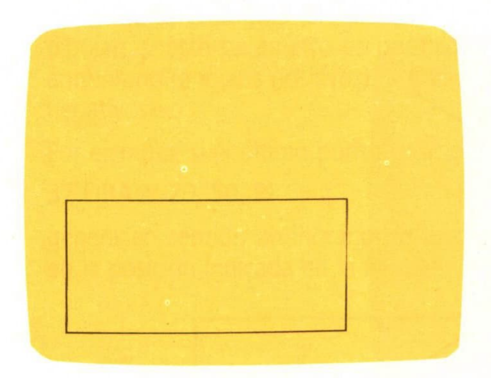
Al ejecutarse el programa la instrucción **10 DRAW 200,** 0 dibuja una recta horizontal desde el punto de partida (0, 0) hasta el punto que se encuentra 200 posiciones a la derecha: (200, 0).

La instrucción **20 DRAW 0, 100** dibuja una recta vertical desde el último punto impreso (200, 0) hasta el punto que se encuentra 100 posiciones hacia arriba: (200, 100).

La instrucción **30 DRAW -200, 0** dibuja una recta horizontal desde el último punto impreso (200, 100) hasta el punto que se encuentra 200 posiciones a la izquierda: (0, 100).

Y, por último, la instrucción **40 DRAW 0, –100** completa el rectángulo, dibujando la recta vertical que une al último punto impreso (0, 100) con el punto que se encuentra 100 posiciones abajo: (0, 0).

En la pantalla se obtiene esta figura:

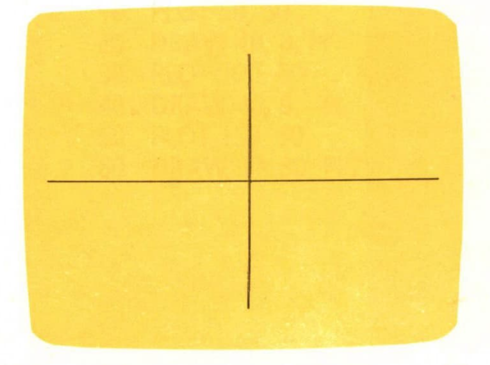


PROGRAMAS RESUELTOS

5. Hacer un programa que dibuje dos ejes coordenados con centro en el punto (128, 88), utilizando la instrucción PLOT para posicional, y la DRAW para trazar rectas.

Solución

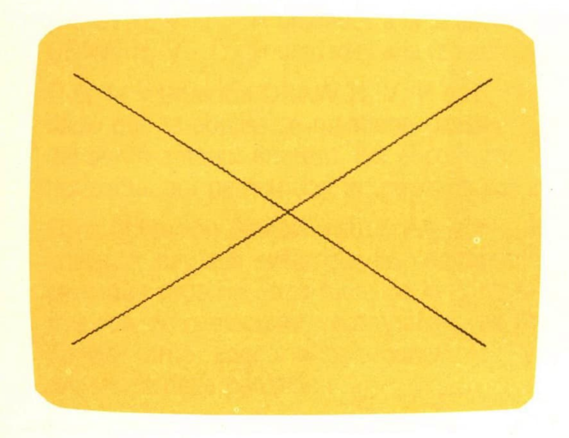
```
10  PLOT 128,0
20  DRAW 0,175
30  PLOT 0,88
40  DRAW 255,0
```



6. Hacer un programa que dibuje las diagonales de la pantalla, es decir, las rectas que unen el punto (0, 0) con (255, 175) y (0, 175) con (255, 0).

Solución

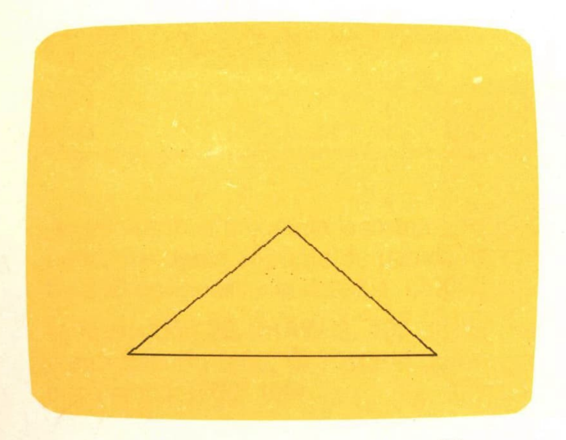
```
10 DRAW 255, 175
20 PLOT 0, 175
30 DRAW 255, –175
```



7. Elaborar un programa que dibuje un triángulo que tenga como vértices los puntos (10, 10), (210, 10) y (110, 100).

Solución

```
10 PLOT 10, 10
20 DRAW 200, 0
30 DRAW –100, 90
40 DRAW –100, –90
```



4. Dibujo de arcos de circunferencia

- Con la instrucción **DRAW** se pueden dibujar arcos de circunferencia, añadiendo un tercer parámetro, N:

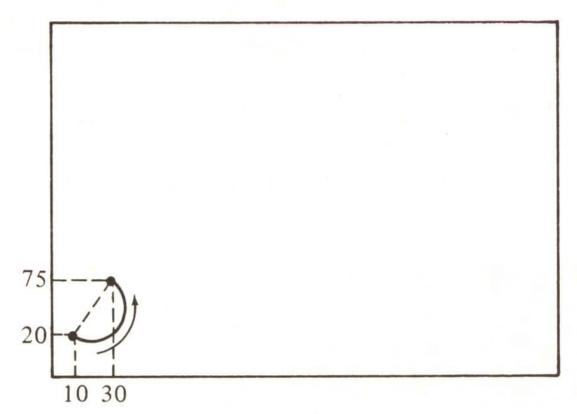
  DRAW H, V, N

  Los extremos del arco corresponden al punto anteriormente impreso o posicionado, y al definido por los valores de los parámetros H y V. El parámetro N indica el ángulo (en radianes) correspondiente al arco que se desea dibujar; si este parámetro es positivo, el arco se va dibujando en sentido antihorario (ángulos positivos), y si es negativo, en sentido horario (ángulos negativos).

  Por ejemplo, si el último punto posicionado es 10, 25, la instrucción

  37 DRAW·20, 50, PI

  genera en sentido antihorario un arco de PI radianes (PI equivale a 180°) en la posición indicada en la figura:



- El programa

```
10  PLOT 50, 50
20  DRAW 40, 0, PI
30  PLOT 100, 50
40  DRAW 40, 0, –PI
50  PLOT 150, 50
60  DRAW 30, 30, PI/2
```

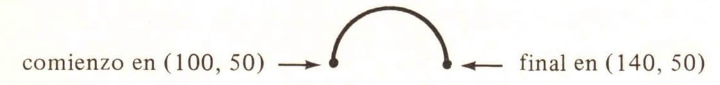
dibujará en pantalla tres arcos:

1. Instrucciones **10** y **20**.



   Como N = PI se obtendrá media circunferencia. El recorrido entre el comienzo y el final se hará en sentido antihorario por ser N positivo.

2. Instrucciones **30** y **40**.

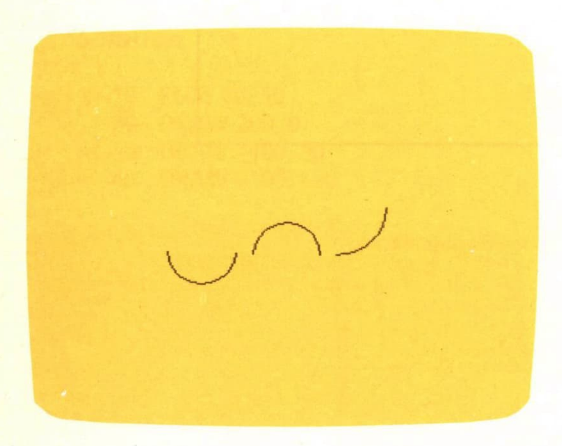


   Como N = –PI, el recorrido se hará en sentido horario y se obtendrá media circunferencia.

3. Instrucciones **50** y **60**.

   El recorrido para dibujar este cuarto de circunferencia se hará en sentido antihorario por ser N = PI/2.

Si se ejecuta el programa se obtiene como resultado el que muestra la fotografía:



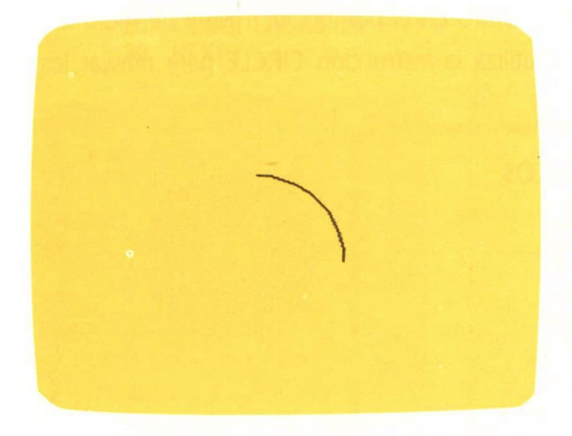
PROGRAMAS RESUELTOS

8. Hacer un programa que dibuje un cuarto de circunferencia cuyos extremos queden definidos por el punto (128, 88), y por el punto si-

tuado a 50 unidades a la izquierda y 50 unidades hacia arriba del anterior. Generar el arco en sentido antihorario.

Solución

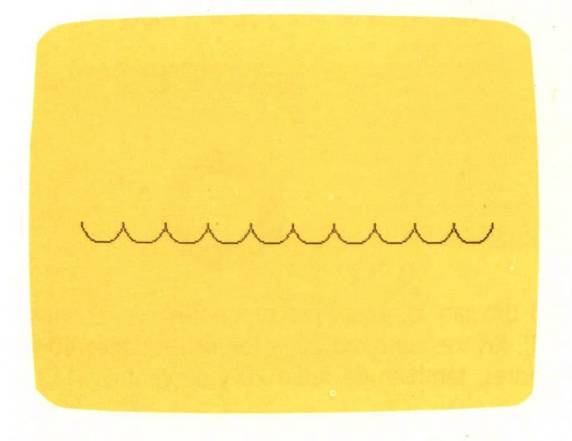
```
10 PLOT 128, 88
20 DRAW –50, 50, PI/2
```



9. Hacer un programa que dibuje diez medias circunferencias que unan los puntos (0, 88) y (250, 88). La primera habrá de ir del (0, 88) al (25, 88), la segunda del (25, 88) al (50, 88), etc.

Solución

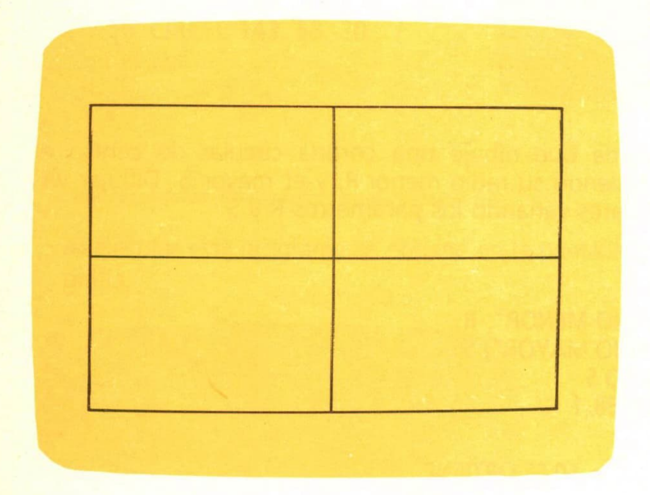
```
10 PLOT 0, 88
20 FOR i = 1 TO 10
30 DRAW 25, 0, PI
40 NEXT i
```



12. Hacer un programa que recuadre toda la pantalla y dibuje dos ejes coordenados con origen en el centro de la pantalla (128, 88).

Solución

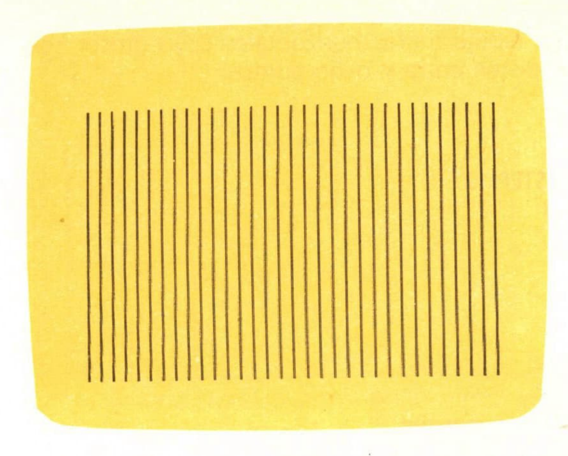
```
  1 REM RECUADRAR TODA LA PANTALLA Y DIBUJAR UNOS EJES
COORDENADOS CENTRADOS EN EL PIXEL 128, 88
 10 FOR X = 0 TO 255
 20 PLOT X, 0
 30 PLOT X, 175
 40 PLOT X, 88
 50 NEXT X
 60 FOR Y = 0 TO 175
 70 PLOT 0, Y
 80 PLOT 255, Y
 90 PLOT 128, Y
100 NEXT Y
```



13. Hacer un programa que dibuje rectas verticales que equidisten entre sí ocho puntos, y que ocupen toda la pantalla.

Solución

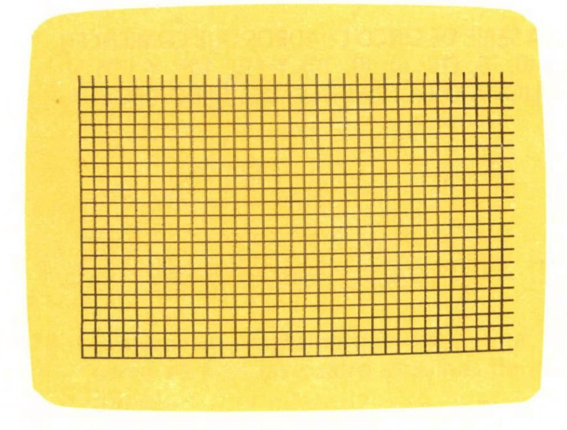
```
 1 REM BARRAS VERTICALES DE 8 EN 8 PUNTOS
10 FOR X = 0 TO 255 STEP 8
20 FOR Y = 0 TO 175
30 PLOT X, Y
40 NEXT Y
50 NEXT X
```



14. Escribir un programa que cuadricule la pantalla por medio de líneas verticales y horizontales.

Solución

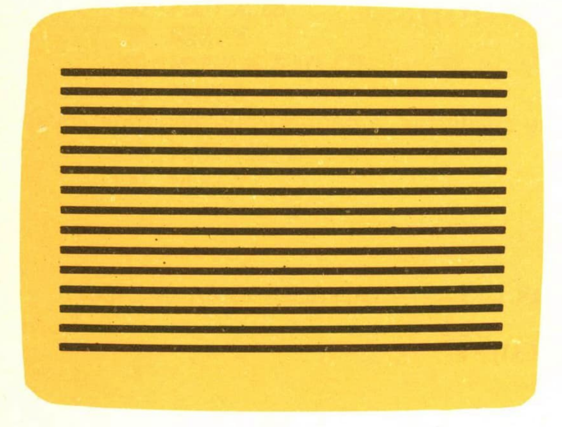
```
  1 REM CUADRICULAR LA PANTALLA VALIENDO CADA LADO 8
PIXELS
 10 FOR Y = 0 TO 175 STEP 8
 20 FOR X = 0 TO 255
 30 PLOT X, Y
 40 NEXT X
 50 NEXT Y
 60 FOR X = 0 TO 255 STEP 8
 70 FOR Y = 0 TO 175
 80 PLOT X, Y
 90 NEXT Y
100 NEXT X
```



15. Hacer un programa que dibuje barras horizontales cuyo grosor sea de cuatro puntos y que disten entre sí ocho puntos.

Solución

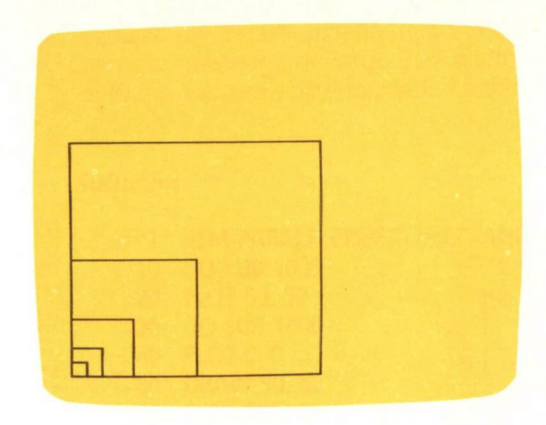
```
10  FOR i = 0 TO 175 STEP 12
20  FOR j = i TO i + 3
30  PLOT 0, j
40  DRAW 255, 0
50  NEXT j
60  NEXT i
```



16. Diseñar un programa que dibuje una serie de cinco cuadrados, siendo el primero el de vértices (0, 0), (10, 0), (0, 10) y (10, 10) y los siguientes vayan duplicando la longitud del lado del cuadrado anterior. (El punto (0, 0) es vértice de todos los cuadrados.)

Solución

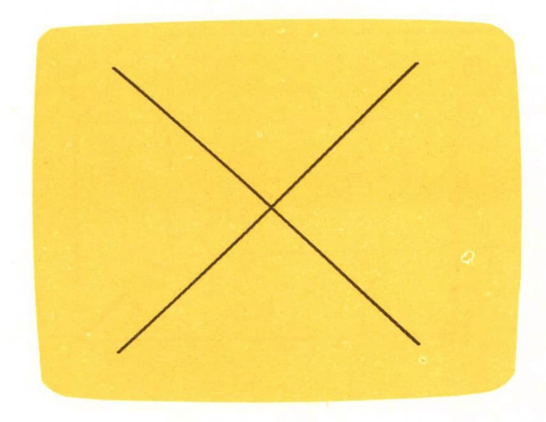
```
   1 REM DIBUJAR UNA SERIE DE CINCO CUADROS QUE COMIENCEN
CON EL DE VERTICES: (0, 0), (10, 0), (0, 10) Y (10, 10); Y LOS SI-
GUIENTES VAYAN DUPLICANDO LA LONGITUD DEL LADO DEL CUA-
DRADO ANTERIOR
  10  LET L = 10
  20  FOR I = 1 TO 5
  30  FOR K = 0 TO L
  40  PLOT K, 0
  50  PLOT K, L
  60  PLOT 0, K
  70  PLOT L, K
  80  NEXT K
  90  LET L = 2 *L
 100  NEXT I
```



17. Hacer un programa que al ejecutarse produzca el efecto de que el punto (128, 88) se dispare según las bisectrices de los cuatro cuadrantes.

Solución

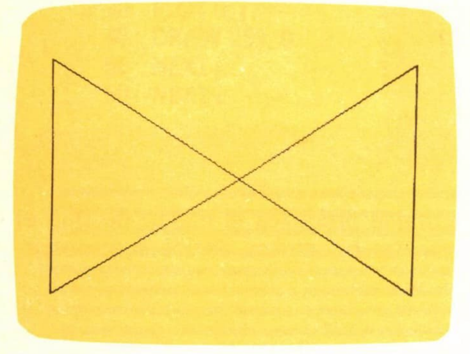
```
 1  REM PUNTO QUE SE DISPARA DESDE EL 128, 88; SEGUN LAS BI-
SECTRICES DE LOS CUATRO CUADRANTES
10  FOR J = 0 TO 87
20  PLOT 128 + J, 88 + J
30  PLOT 128 + J, 88 − J
40  PLOT 128 − J, 88 + J
50  PLOT 128 − J, 88 − J
60  NEXT J
```



18. Elaborar un programa que dibuje una "pajarita de vestir" que tenga como vértices las cuatro esquinas de la pantalla (solamente su contorno).

Solución

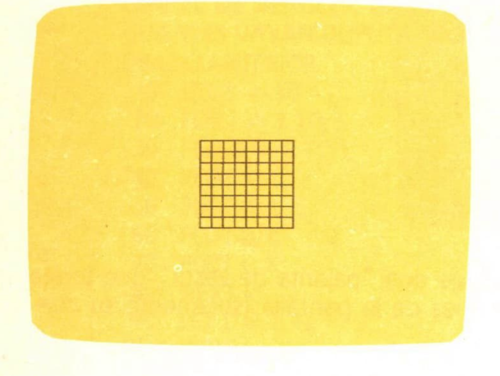
```
 1 REM PAJARITA
10 DRAW 255, 175
20 DRAW 0, -175
30 DRAW -255, 175
40 DRAW 0, -175
```



19. Hacer un programa que dibuje un tablero de ajedrez, teniendo en cuenta que el lado de cada cuadrado ha de medir 8 y que la esquina inferior izquierda del mismo debe ser el punto (8 * 11,8 * 10).

Solución

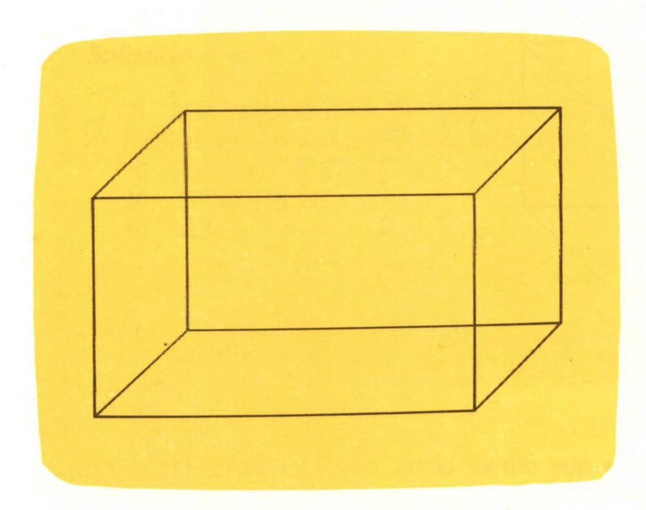
```
10 FOR i = 0 TO 8
20 PLOT 8 * 11, 8 * 10 + 8 * i
30 DRAW 8 * 8, 0
40 NEXT i
50 FOR i = 0 TO 8
60 PLOT 8 * 11 + 8 * i, 8 * 10
70 DRAW 0, 8 * 8
80 NEXT i
```



20. Dibujar un paralelepípedo recto rectangular (ortoedro) tal que su cara frontal tenga de largo 205 y alto 125 y uno de sus vértices sea el (0, 0). La cara posterior debe contener el vértice (50, 50).

Solución

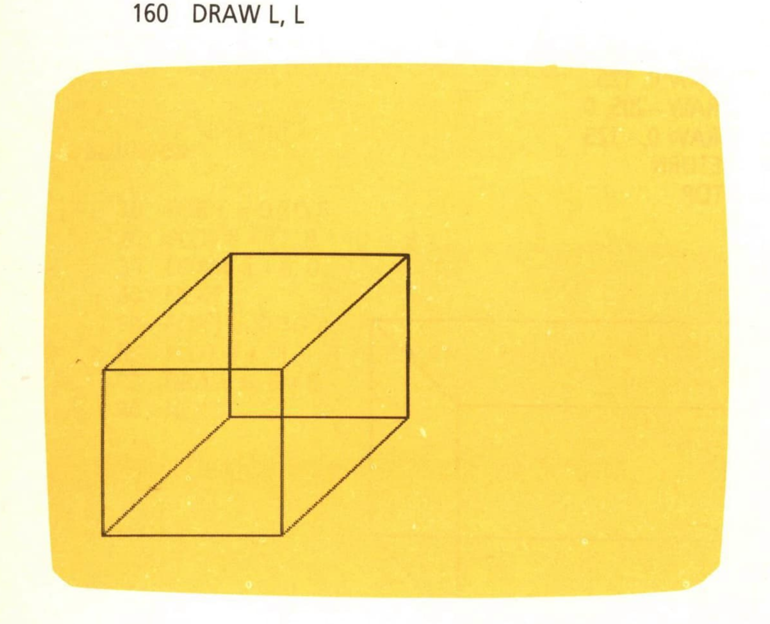
```
   1 REM PARALELEPIPEDO RECTANGULAR
  10 GO SUB 1000
  20 PLOT 50, 50
  30 GO SUB 1000
  40 PLOT 0, 0
  50 DRAW 50, 50
  60 PLOT 205, 0
  70 DRAW 50, 50
  80 PLOT 205, 125
  90 DRAW 50, 50
 100 PLOT 0, 125
 110 DRAW 50, 50
 120 GO TO 1070
1000 REM SUBRUTINA RECTANGULO
1020 DRAW 205,0
1030 DRAW 0, 125
1040 DRAW -205, 0
1050 DRAW 0, -125
1060 RETURN
1070 STOP
```



21. Hacer un programa que dibuje un cubo de lado 80 y que uno de sus vértices sea el punto (0, 0).

Solución

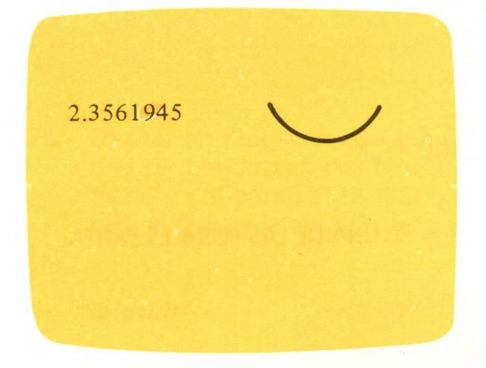
```
  1 REM DIBUJAR UN CUBO DE LADO
80 Y QUE UNO DE SUS VERTICES SEA
EL PUNTO (0, 0)
 10 DRAW 80, 0
 20 DRAW 0, 80
 30 DRAW -80, 0
 40 DRAW 0, -80
 50 LET L = SQR (80 ↑ 2/2)
 60 DRAW L, L
 70 DRAW 80, 0
 80 DRAW 0, 80
 90 DRAW -80, 0
100 DRAW 0, -80
110 PLOT 80, 0
120 DRAW L, L
130 PLOT 80, 80
140 DRAW L, L
150 PLOT 0, 80
160 DRAW L, L
```



22. Hacer un programa que dibuje arcos, desde el punto (113, 175) al (143, 175), que vayan variando de 0 a 2 *PI, con paso de PI/8 (imprimir el valor del ángulo). Además, la imagen debe permanecer en la pantalla durante una pausa de 50 cada vez que se dibuje un nuevo arco y, a continuación, se debe borrar la pantalla.

Solución

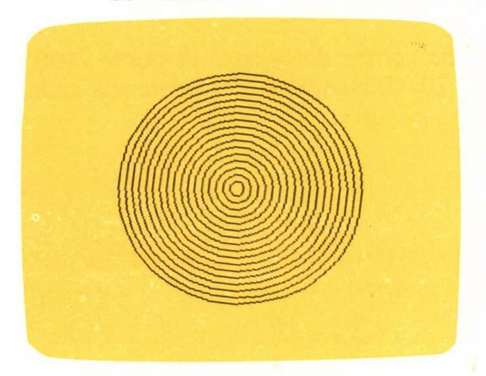
```
 1 REM HACER UN PROGRAMA QUE DIBUJE ARCOS
10 FOR i = 0 TO 2 * PI STEP PI/8
20 PRINT i
30 PLOT 113, 175
40 DRAW 30, 0, i
50 PAUSE 50
60 CLS
70 NEXT i
```



23. Hacer un programa que dibuje una diana centrada en el punto (128, 88), de tal manera que los radios de las circunferencias concéntricas vayan aumentando de 5 en 5.

Solución

```
 1 REM DIBUJAR UNA DIANA CENTRADA EN 128, 88 Y QUE LOS
RADIOS VAYAN AUMENTANDO DE 5 EN 5
10 FOR I = 5 TO 87 STEP 5
20 CIRCLE 128, 88, I
30 NEXT I
```

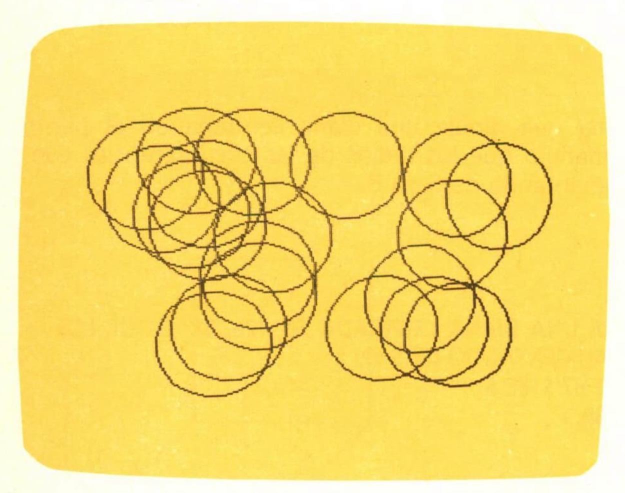


24. Diseñar un programa que dibuje N circunferencias con centros aleatorios y radio R. (Las circunferencias deben quedar dentro de la pantalla.)

Solución

```
 1 REM DIBUJA N CIRCUNFERENCIAS CON CENTROS ALEATORIOS
Y RADIO R
10 INPUT "CUANTAS CIRCUNFERENCIAS"; N
20 INPUT "RADIO (MAXIMO 87)"; R
30 RANDOMIZE
40 LET X = 255 - 2 *R
50 LET Y = 175 - 2 *R
60 FOR I = 1 TO N
70 CIRCLE R + RND *X, R + RND *Y, R
80 NEXT I
```

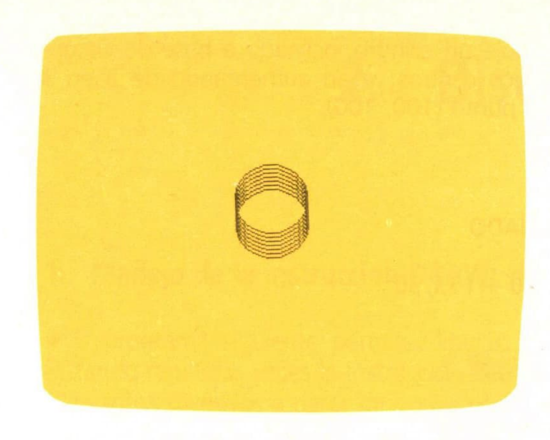
POR EJEMPLO PARA N = 20 Y R = 30 UNA DE LAS POSIBLES PANTALLAS ES



25. Dibujar un cilindro que tenga de centro de la base el punto (100, 100), de radio, 25 y de altura, 20. (Dibujar una circunferencia cada tres puntos de altura.)

Solución

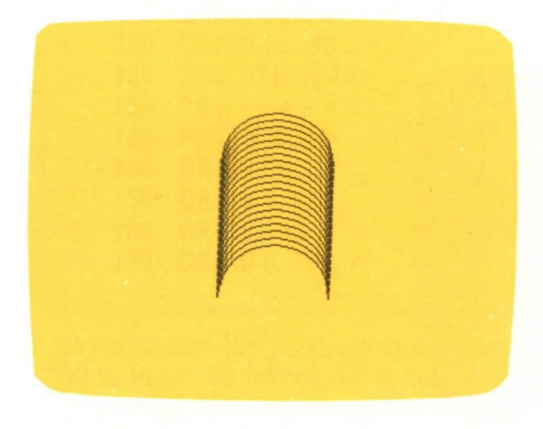
```
 1 REM CICLINDRO
10 FOR I = 0 TO 20 STEP 3
20 CIRCLE 100, 100 + I, 25
30 NEXT I
```



26. Diseñar un programa que dibuje un cilindro tal que el radio y la altura se introduzcan por el teclado, estando el centro de la base situado en el punto (128, 38). ¿Qué valores máximos se pueden dar para el radio y la altura?

Solución

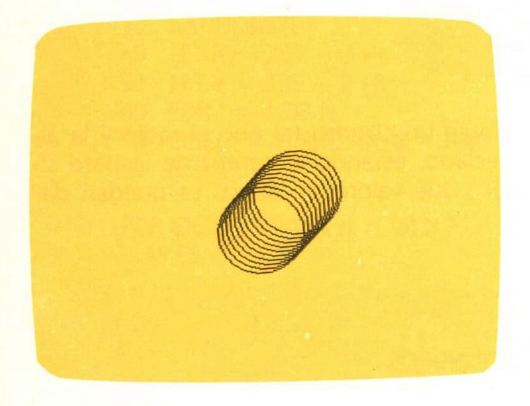
```
1 REM DIBUJAR MEDIOS CILINDROS
CON RADIO Y ALTURA LA QUE SE
INTRODUZCA Y COMO CENTRO DE LA BASE
EL PUNTO (128, 38).
10 INPUT "RADIO (MAXIMO 38)"; R
20 INPUT "ALTURA (MAXIMO 98)"; H
25 FOR I = 0 TO H STEP 5
27 PLOT 128 -R, 38 + I
28 DRAW 2 * R, 0, -PI
40 NEXT I
PARA R = 38 y H = 98 SE TIENE
```



27. Hacer un programa que dibuje un cilindro inclinado a base de circunferencias de radio 30 y cuyos centros vayan aumentando de 3 en 3 puntos, comenzando por el punto (100, 100).

Solución

```
 1 REM CILINDRO INCLINADO
10 FOR I = 1 TO 10
20 CIRCLE 100 + I *3, 100 + I *3, 30
30 NEXT I
```



28. Hacer un programa que dibuje un cono por medio de circunferencias, cuyos radios R varían de 4 a 50 con incrementos de 4 y sus centros comiencen en el punto (80, 10) y se incremente R y 2 * R.

Solución

```
 1 REM CONO
10 FOR R = 4 TO 50 STEP 4
20 CIRCLE 80 + R, 10 + 2 * R, R
30 NEXT R
```



2. Diseño

1. Manejo de la instruccion DRAW: dibujo de una casa

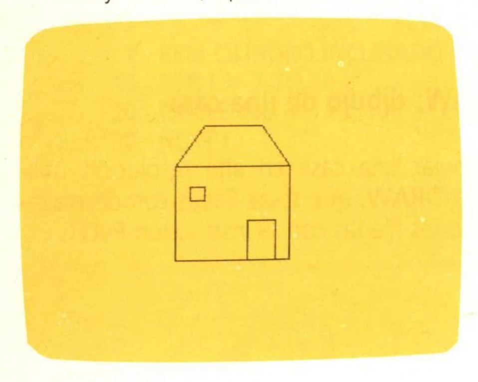
- El programa siguiente permite dibujar una casa en alta resolución utilizando repetidas veces la instrucción **DRAW**, que traza líneas con desplazamientos sucesivos a partir de posiciones fijadas con la instrucción **PLOT**.

```
DIBUJO DE UNA CASA
  5  REM TEJADO
  6  REM * * * * *
 10  PLOT 100, 80
 20  DRAW 20, 30
 30  DRAW 40, 0
 40  DRAW 20, -30
 50  DRAW - 80, 0
 55  REM PAREDES Y SUELO
 56  REM * * * * * * * * * * *
 60  DRAW 0, -70
 70  DRAW 80, 0
 80  DRAW 0, 70
 85  REM PUERTA
 86  REM * * * * *
 90  PLOT 170, 10
100  DRAW 0, 30
110  DRAW -20, 0
120  DRAW 0, -30
125  REM VENTANA
126  REM * * * * * *
130  PLOT 110, 65
140  DRAW 10, 0
150  DRAW 0, -10
160  DRAW -10, 0
170  DRAW 0, 10
```

La instrucción **10** fija un punto, a partir del cual en las instrucciones **20 a 50** se trazan los tramos rectos que constituyen el tejado. Las instrucciones **60, 70** y **80** dibujan las paredes y el suelo.

Finalmente, se dibujan la puerta (instrucciones **90** a **120**) y la ventana (instrucciones **140** a **170**).

Para obtener el dibujo que produce este programa pulsamos las teclas **RUN** y **ENTER,** apareciendo en la pantalla lo que muestra la fotografía.



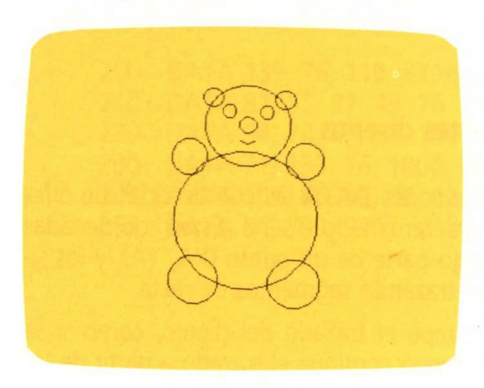
2. Manejo de la instrucción circle: dibujo de un oso.

- Con el uso de sucesivas instrucciones **CIRCLE** se puede también conseguir diversas figuras. Como ejemplo, transcribimos el siguiente programa que dibuja un oso en la pantalla.

```
DIBUJO DE UN OSO
 10  CIRCLE 120, 70, 50
 20  CIRCLE 120, 140, 30
 30  CIRCLE 85, 25, 18
 40  CIRCLE 155, 25, 18
 50  CIRCLE 80, 115, 13
 60  CIRCLE 160, 115, 13
 70  CIRCLE 95, 160, 7
 80  CIRCLE 145, 160, 7
 90  CIRCLE 107, 150, 5
100  CIRCLE 133, 150, 5
110  CIRCLE 120, 140, 6
120  PLOT 115, 128
130  DRAW 5, –2
140  DRAW 5,2
```

Con las instrucciones **10** a **110** se dibuja el cuerpo, cabeza, patas delanteras y traseras, orejas, ojos y nariz del oso. Únicamente se usan instrucciones **DRAW** (**130** y **140**) para dibujar la boca.

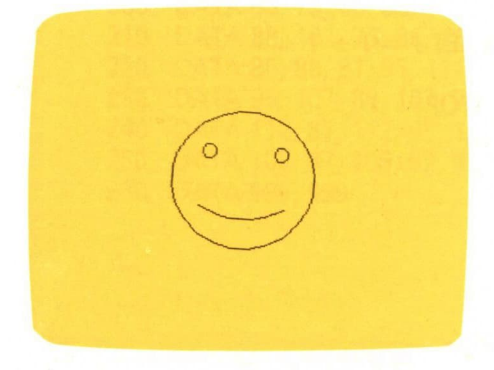
Ejecutando el programa (pulsar **RUN** y **ENTER**) se obtiene lo que muestra la fotografía.



3. Dibujo de una cara sonriente

- A la instrucción **DRAW** se le puede dar un tercer argumento, que en este caso nos permite dibujar la boca de una cara sonriente.

```
CARA SONRIENTE
  10  CIRCLE 128, 88, 50
  20  CIRCLE 103, 108, 5
  30  CIRCLE 153, 108, 5
  40  PLOT 98, 68
  50  DRAW 60, 0, PI/3
```



Esta misma cara se puede convertir en triste sustituyendo la instrucción **50** por

50 DRAW 60, 0, –PI/3

4. Programa que realiza diferentes diseños

- El programa siguiente lee en instrucciones **DATA** las coordenadas de diferentes puntos que constituyen un determinado diseño. Estas coordenadas se almacenan en una tabla. El dibujo parte de un punto (XA, YA), y los siguientes se obtienen a partir de él, trazando segmentos de recta.

  Si el dato leído es 1000 se interrumpe el trazado del dibujo, como si se tratara de un lápiz que se levanta; luego continúa el trazado a partir de un punto determinado.

  Cuando las coordenadas del punto guardadas en la instrucción **DATA** valen 999 se da por terminado el dibujo.

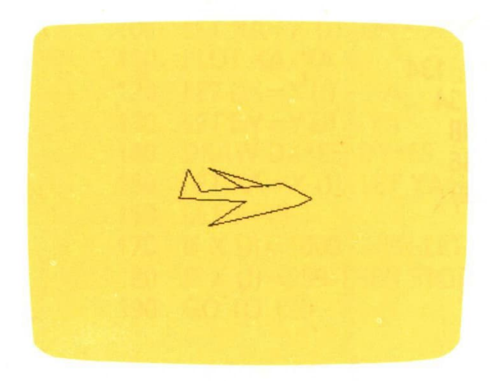
```
PROGRAMA QUE REALIZA DIFERENTES DISEÑOS
 10  DIM X (100): DIM Y (100)
 20  LET J = 1
 30  FOR I = 1 TO 100
 40  READ X (I), Y (I)
 50  IF X (I) = 999 THEN GO TO 100
 60  NEXT I
100  LET XA = X(J): LET YA = Y(J)
110  PLOT XA, YA
120  LET DX = X(J) – XA
130  LET DY = Y(J) – YA
140  DRAW DX, DY
150  LET XA = X(J): LET YA = Y(J)
160  LET J = J + 1
170  IF X (J) = 1000 THEN LET J = J + 1
GO TO 100
180  IF X (J) = 999 THEN STOP
190  GO TO 120
200  DATA...
```

- **Dibujo de un avión**

  Si al programa anterior le añadimos las siguientes instrucciones DATA

```
200  DATA 139, 76, 119, 87, 64
210  DATA 81, 55, 97, 49, 76
220  DATA 94, 74, 69, 56, 130
230  DATA 71, 139, 76, 1000, 1000
240  DATA 109, 85, 69, 98, 92
250  DATA 84, 999, 999
```

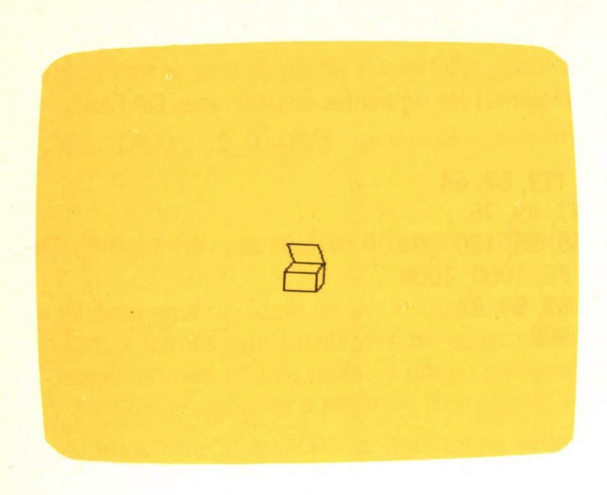
se obtiene el dibujo de un avión, como el que muestra la fotografía.



- **Dibujo de una caja**

  Con las siguientes instrucciones DATA se obtiene una caja.

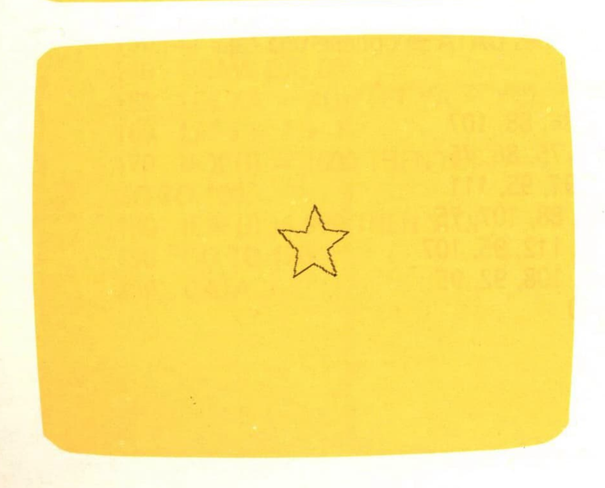
```
200  DATA 86, 75, 86, 88, 107
210  DATA 88, 107, 75, 86, 75
220  DATA 86, 88, 91, 95, 111
230  DATA 95, 107, 88, 107, 75
240  DATA 112, 81, 112, 95, 107
250  DATA 108, 87, 108, 92, 95
260  DATA 999, 999
```



- **Dibujo de una estrella**

Con estas instrucciones **DATA** se obtiene una estrella.

```
200  DATA 110, 105, 117, 89, 134
210  DATA 89, 122, 75, 125, 54
220  DATA 110, 67, 95, 54, 98
230  DATA 73, 86, 84, 104, 86
240  DATA 109, 105, 999, 999
```



5. Dibujo con escala

- El tamaño de los dibujos se puede cambiar modificando la escala, añadiendo al programa anterior la instrucción **5** y modificando la **140**.

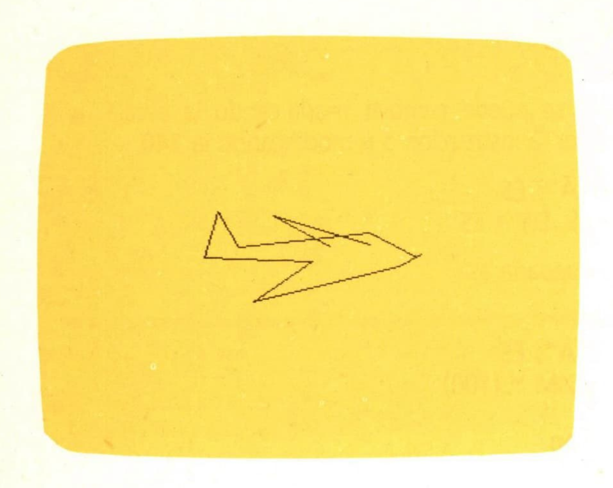
```
  5 INPUT "ESCALA"; ES
140 DRAW DX * ES, DY * ES
```

El programa modificado quedaría así:

```
  5 INPUT "ESCALA"; ES
 10 DIM X (100): DIM Y (100)
 20 LET J=1
 30 FOR I=1 TO 100
 40 READ X (I), Y (I)
 50 IF X (I)=999 THEN GO TO 100
 60 NEXT I
100 LET XA=X (J): LET YA=Y (J)
110 PLOT XA, YA
120 LET DX=X (J) -XA
130 LET DY=Y (J) -YA
140 DRAW DX*ES, DY*ES
150 LET XA=X (J): LET YA=Y (J)
160 LET J=J+1
170 IF X (J)=1000 THEN LET J=J+1 : GO TO 100
180 IF X (J)=999 THEN STOP
190 GO TO 120
```

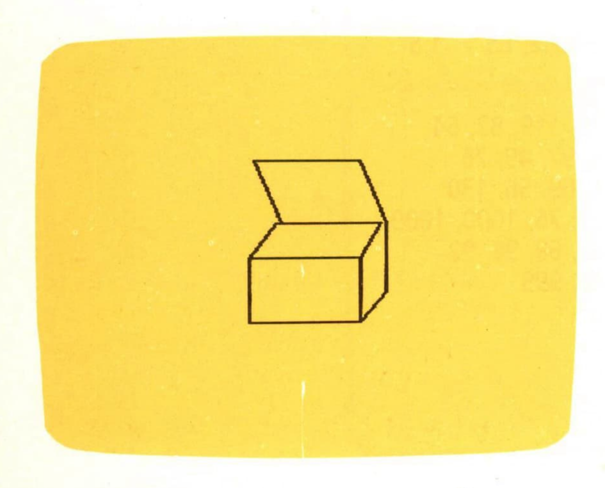
- **Dibujo del avión con escala, ES = 1,5**

```
200 DATA 139, 76, 119, 87, 64
210 DATA 81, 55, 97, 49, 76
220 DATA 94, 74, 69, 56, 130
230 DATA 71, 139, 76, 1000, 1000
240 DATA 109, 85, 69, 98, 92
250 DATA 84, 999, 999
```



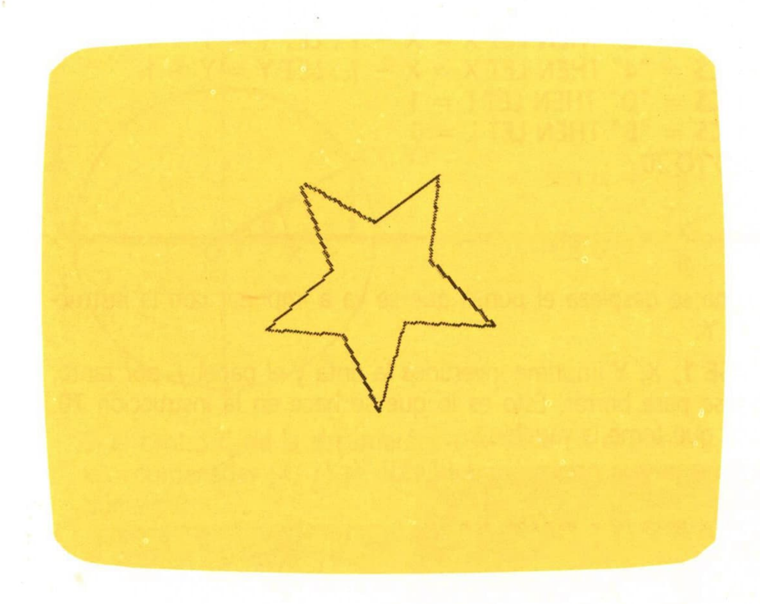
- **Dibujo de la caja con escala, ES = 3**

```
200  DATA 86, 75, 86, 88, 107
210  DATA 88, 107, 75, 86, 75
220  DATA 86, 88, 91, 95, 111
230  DATA 95, 107, 88, 107, 75
240  DATA 112, 81, 112, 95, 107
250  DATA 108, 87, 108, 92, 95
260  DATA 999, 999
```



- **Dibujo de la estrella con escala, ES = 2**

```
200  DATA 110, 105, 117, 89, 134
210  DATA 89, 122, 75, 125, 54
220  DATA 110, 67, 95, 54, 98
230  DATA 73, 86, 84, 104, 86
240  DATA 109, 105, 999, 999
```



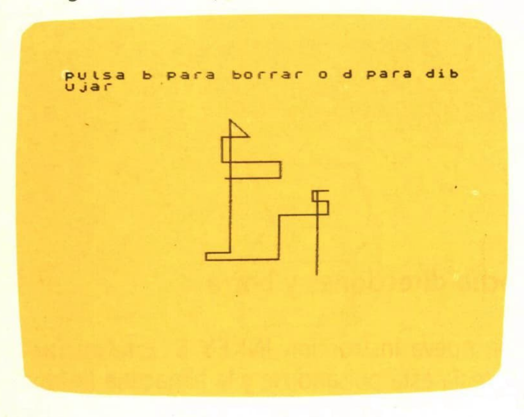
6. Programa que dibuja en ocho direcciones y borra

- El programa siguiente utiliza una nueva instrucción **INKEY $**. Esta instrucció lee el teclado buscando, qué tecla está pulsándose y la almacena en una variable de cadena. También se utiliza la instrucción **PAUSE 0** que interrumpe la ejecución hasta que se pulse una tecla.

```
  5 PRINT "PULSA B PARA BORRAR O D PARA DIBUJAR"
  7 PAUSE 0
 10 LET X = 100: LET Y = 100: LET L = 0
 20 LET C$ = INKEY$ : IF C$ <>" " THEN GO TO 60
 30 PLOT X, Y
 40 PLOT INVERSE 1; X, Y
 50 GO TO 20
 60 IF L = 0 THEN PLOT X, Y
 70 IF L = 1 THEN PLOT INVERSE 1; X, Y
 80 IF C$ = "5" THEN LET X = X - 1
 90 IF C$ = "8" THEN LET X = X + 1
100 IF C$ = "6" THEN LET Y = Y - 1
110 IF C$ = "7" THEN LET Y = Y + 1
120 IF C$ = "1" THEN LET X = X + 1 : LET Y = Y - 1
130 IF C$ = "2" THEN LET X = X + 1 : LET Y = Y + 1
140 IF C$ = "3" THEN LET X = X - 1 : LET Y = Y - 1
150 IF C$ = "4" THEN LET X = X - 1 : LET Y = Y + 1
200 IF C$ = "D" THEN LET L = 1
210 IF C$ = "B" THEN LET L = 0
220 GO TO 20
```

En el programa se desplaza el punto que se va a imprimir con la instrucción **PLOT X, Y.**

PLOT INVERSE 1; X, Y imprime invertidos la tinta y el papel y, por tanto, puede utilizarse para borrar. Esto es lo que se hace en la instrucción **70,** según el valor que tome la variable L.

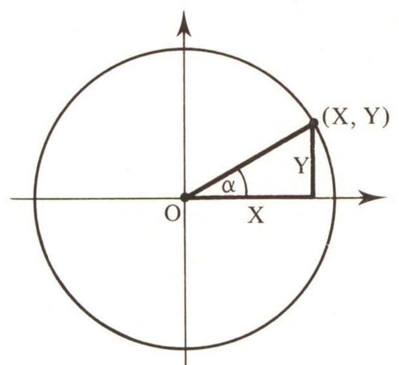


3. Dibujos a partir de la circunferencia

1. Dibujo de una circunferencia «punto a punto»

Una circunferencia se puede dibujar directamente con la instrucción CIRCLE. Pero puede interesar dibujarla ***punto a punto,*** para lo cual es necesario determinar previamente las coordenadas de uno de los puntos de dicha circunferencia.

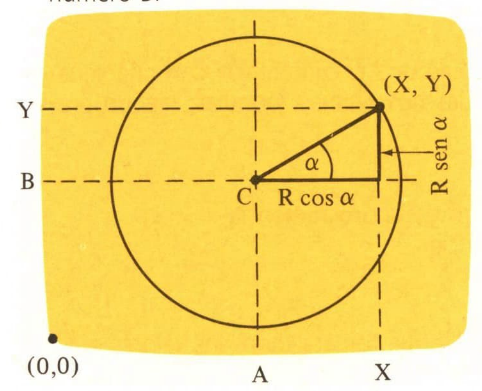
Suponiendo que el radio de la circunferencia es R y que el origen de coordenadas coincide con su centro, las coordenadas (X, Y) de un punto cualquiera de la misma, se obtienen partiendo de la definición de seno y coseno.



$$\text{sen}\ \alpha = \frac{Y}{R} \Rightarrow Y = R\ \text{sen}\ \alpha$$

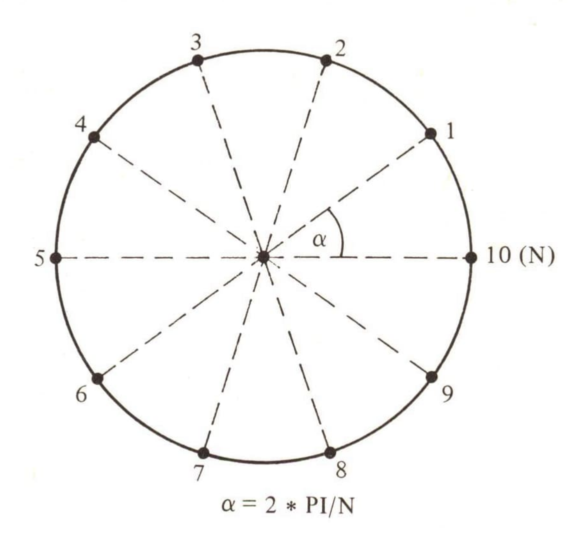
$$\cos \alpha = \frac{X}{R} \Rightarrow X = R \cos \alpha$$

Si el centro C de la circunferencia se sitúa en el pixel A, B de la pantalla, las coordenadas (X, Y) se obtendrán sumando a X el número A y a Y el número B.



$$X = A + R \cos \alpha$$
$$Y = B + R\ \text{sen}\ \alpha$$

Si decidimos dibujar la circunferencia con N puntos, tendremos que fijar previamente el valor del ángulo α necesario para pasar de un punto al siguiente. Este valor se calcula teniendo en cuenta que el ángulo completo correspondiente a toda la circunferencia es 360°, que equivale a 2 π radianes; luego si dibujamos N puntos, tendremos:



$\alpha = 2 * PI/N$

Es decir, el ángulo correspondiente al punto 1 valdrá 2 * PI/N * 1, el correspondiente al punto 2, 2 * PI/N * 2, ..., y el correspondiente al punto N, 2 * PI/N * N = 2 * PI.

Luego, las coordenadas de los sucesivos puntos de la circunferencia se podrán determinar mediante las siguientes fórmulas:

X = A + R * COS (2 * PI/N * I)
Y = B + R * SIN (2 * PI/N * I)

siendo I = 1, 2, ..., N.

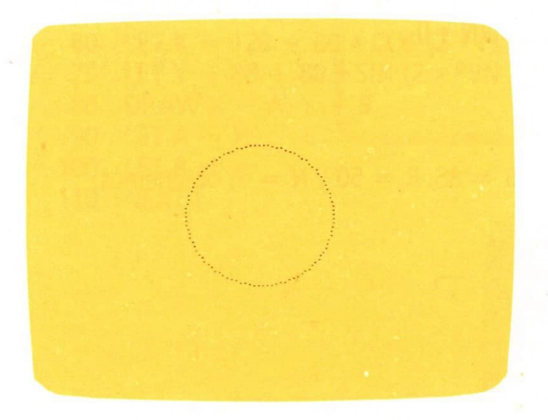
Para elaborar el programa que dibuje una circunferencia con una serie de puntos, hay que tener en cuenta las dos últimas fórmulas, fijando previamente los siguientes parámetros:

A, B: coordenadas del centro de la circunferencia
R: radio de la circunferencia
N: número de puntos

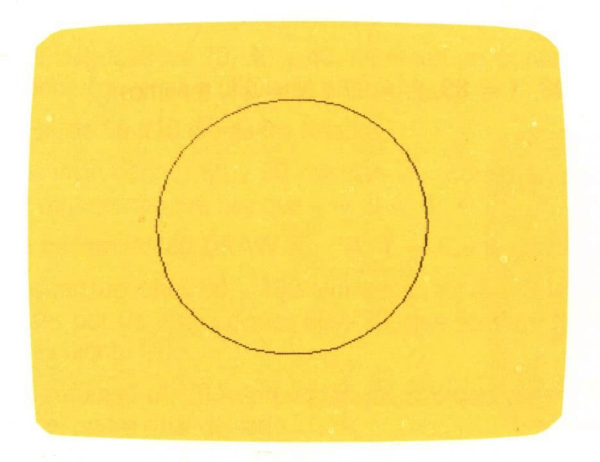
Además, hay que definir la variable I, que tomará todos los valores enteros desde 1 a N, ambos inclusive. Así se obtiene el siguiente programa.

```
10  INPUT "CENTRO"; A,B
20  INPUT "RADIO"; R
30  INPUT "NUMERO DE PUNTOS"; N
40  FOR I = 1 TO N
50  LET X = A + R * COS (2 * PI/N * I)
60  LET Y = B + R * SIN (2 * PI/N * I)
70  PLOT X,Y
80  NEXT I
```

Si lo ejecutamos para A = 128, B = 88, R = 50 y N = 100, obtenemos:



Si lo ejecutamos para A = 100, B = 100, R = 75 y N = 200, tenemos:



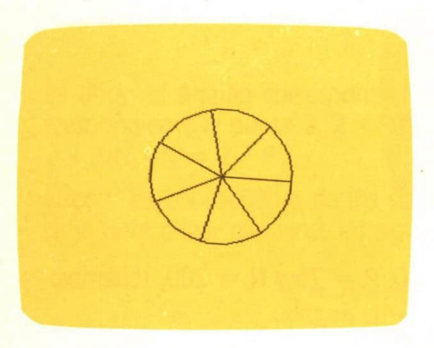
2. División de un círculo en N partes iguales

Para dividir una circunferencia en N partes iguales, la dibujaremos previamente con una instrucción **CIRCLE** y, a continuación, uniremos su centro con cada uno de los N puntos; así se obtendrán N sectores iguales.

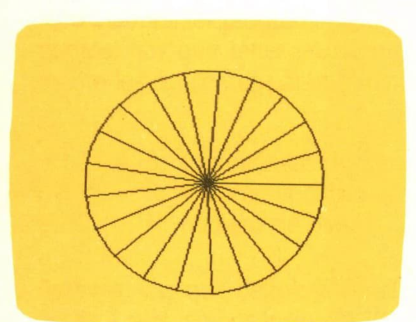
El programa que realiza esta división es el siguiente:

```
 10 INPUT "CENTRO"; A,B
 20 INPUT "RADIO"; R
 30 INPUT "PARTES"; N
 40 CIRCLE A,B,R
 50 FOR I = 1 TO N
 60 PLOT A,B
 70 LET X = R * COS (2 * PI/N * I)
 80 LET Y = R * SIN (2 * PI/N * I)
 90 DRAW X,Y
100 NEXT I
```

Si lo ejecutamos para A = 128, B = 88, R = 50 y N = 7, obtenemos:



Si lo ejecutamos para A = 128, B = 88, R = 87 y N = 21, tenemos:



3. Polígonos regulares

La división de la circunferencia en N partes iguales sugiere la obtención de polígonos regulares de N lados inscritos en la misma.

El siguiente programa dibuja polígonos regulares pero no la circunferencia en la que están inscritos. Esta circunferencia tiene su centro en el pixel 128, 88 y su radio es 80.

```
 10 INPUT "NUMERO DE LADOS"; N
 20 LET A = 128 + 80
 30 LET B = 88
 40 PLOT A, B
 50 FOR I = 1 TO N
 60 LET X = 128 + 80 * COS (2 * PI/N * I)
 70 LET Y = 88 + 80 * SIN (2 * PI/N * I)
 80 DRAW X - A, Y - B
 90 LET A = X
100 LET B = Y
110 NEXT I
```

Con la instrucción **10** se introduce el número de lados del polígono.

Las instrucciones **20, 30** y **40** imprimen un punto en el pixel 128 + 80, 88, donde comienza a dibujarse el polígono.

El bucle **50-110** dibuja los N lados.

Las instrucciones **60** y **70** calculan las coordenadas del siguiente punto de la circunferencia que hay que unir al A, B.

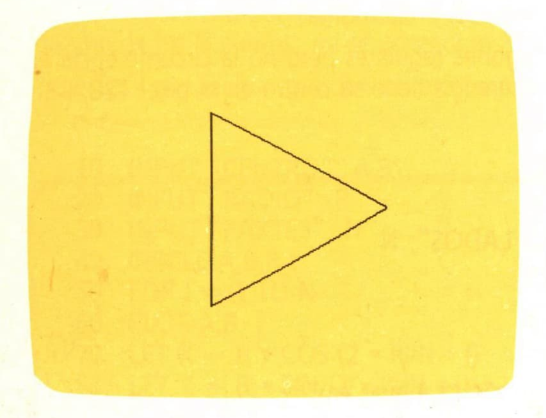
La instrucción **80 DRAW X − A, Y − B** une los pixels A, B y X, Y.

Las instrucciones **90** y **100** almacenan en A y B las coordenadas X, Y, calculadas por las instrucciones **60** y **70,** que serán el pixel de partida para dibujar el siguiente lado.

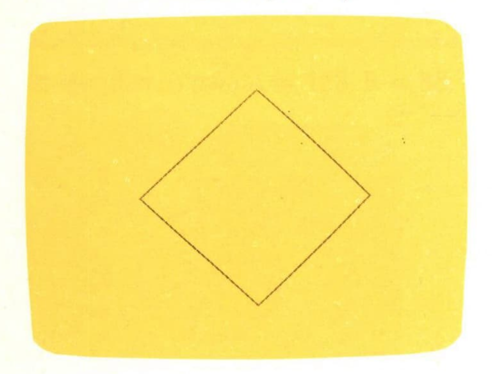
En resumen (A, B), representan las coordenadas del origen de un lado y (X, Y), el punto final del lado.

Si lo ejecutamos, tenemos

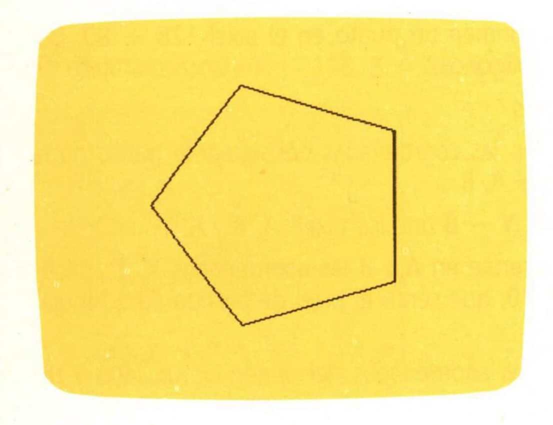
TRIÁNGULO (N = 3)



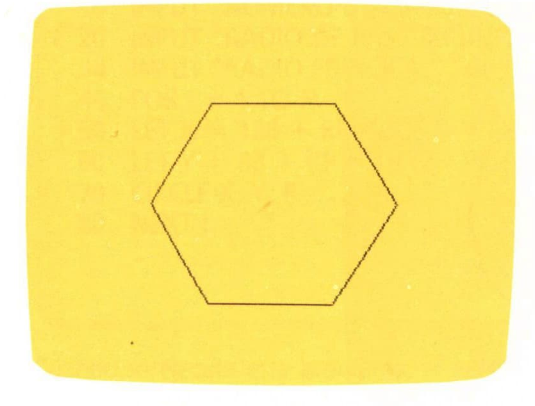
CUADRADO (N = 4)



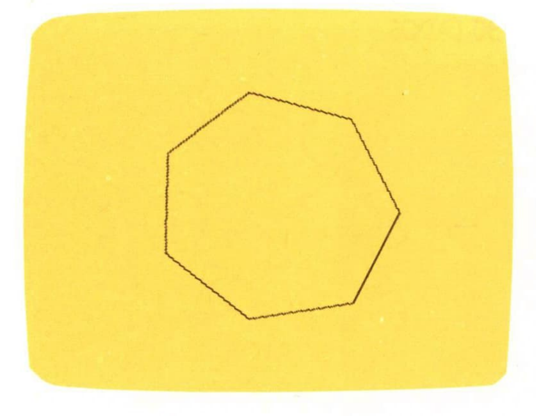
PENTÁGONO (N = 5)



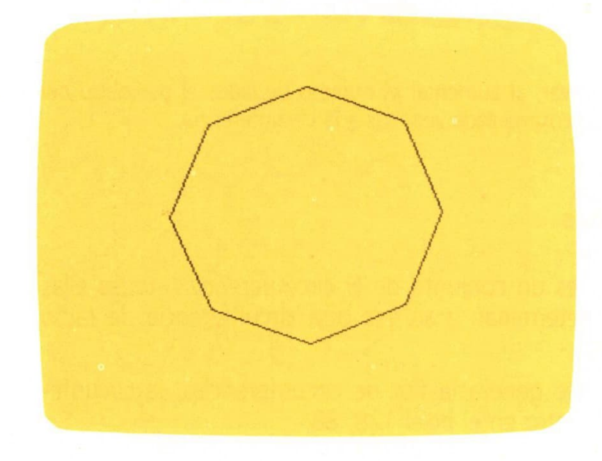
HEXÁGONO (N = 6)



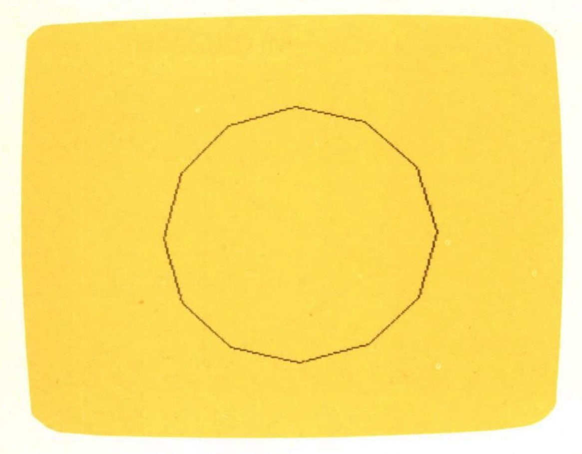
HEPTÁGONO (N = 7)



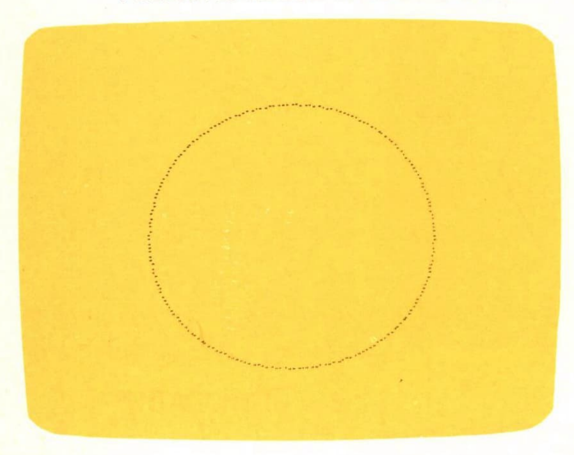
OCTÓGONO (N = 8)



DODECÁGONO (N = 12)



POLÍGONO REGULAR DE 30 LADOS



Nota.—Como se puede observar, al aumentar el número de lados el perímetro del polígono regular se aproxima cada vez más a la circunferencia.

4. Flor de circunferencias

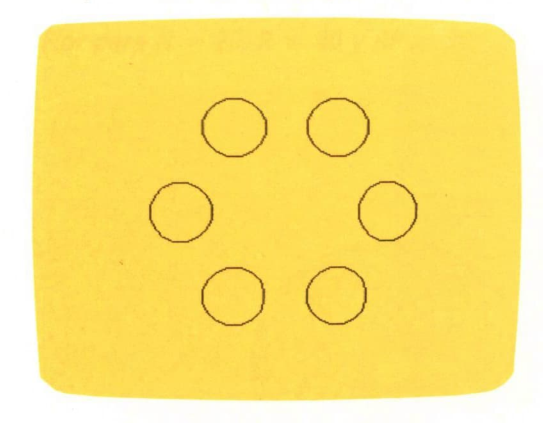
La ***flor de circunferencias*** es un conjunto de N circunferencias, todas ellas de radio R, cuyos centros determinan a su vez otra circunferencia de radio RP, llamado radio principal.

En el programa siguiente, que genera la flor de circunferencias, la circunferencia de radio RP tiene su centro en el pixel 128, 88.

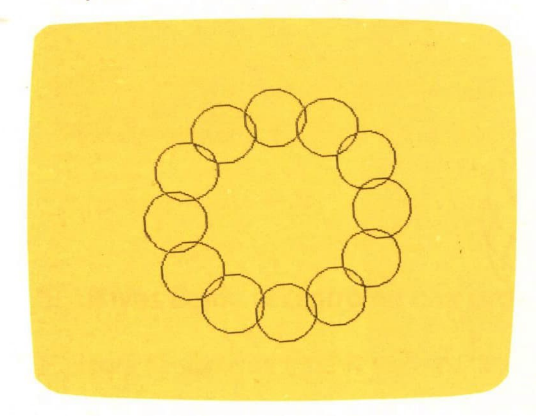
```
10  INPUT "NUMERO DE CIRCUNFERENCIAS"; N
20  INPUT "RADIO DE LAS CIRCUNFERENCIAS"; R
30  INPUT "RADIO PRINCIPAL"; RP
40  FOR I = 1 TO N
50  LET X = 128 + RP * COS (2 * PI/N * I)
60  LET Y = 88 + RP * SIN (2 * PI/N * I)
70  CIRCLE X, Y, R
80  NEXT I
```

Cuando se ejecuta este programa, al introducir los datos hay que evitar que el dibujo se salga de la pantalla, para lo cual debe cumplirse $R + RP \leqslant 87$.

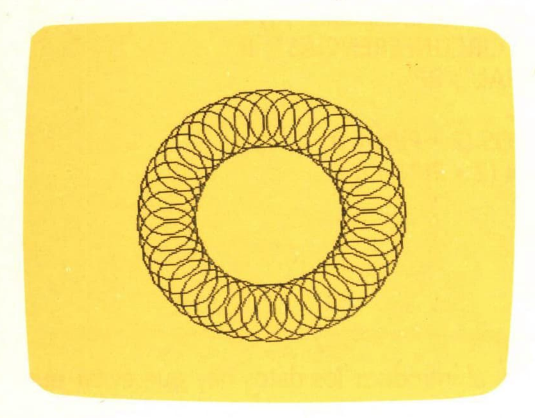
Flor para N = 6, R = 20 y RP = 67



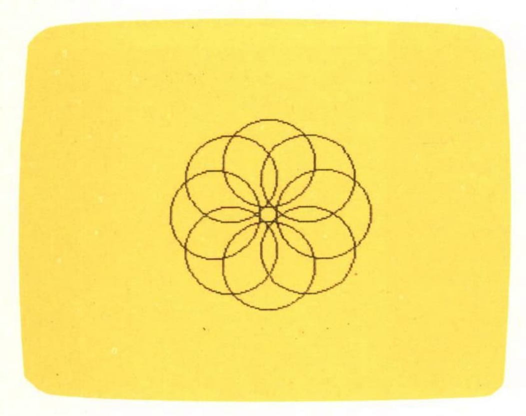
Flor para N = 12, R = 20 y RP = 67



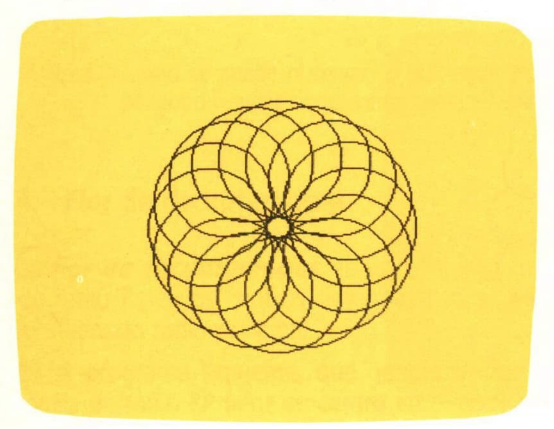
Flor para N = 40, R = 20 y RP = 67



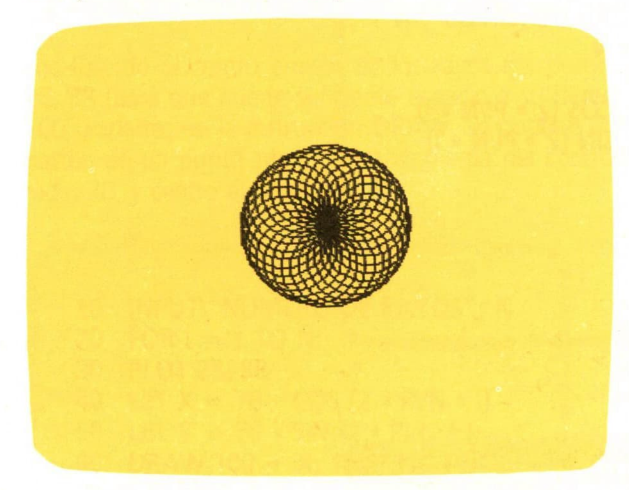
Flor para N = 8, R = 30 y RP = 35



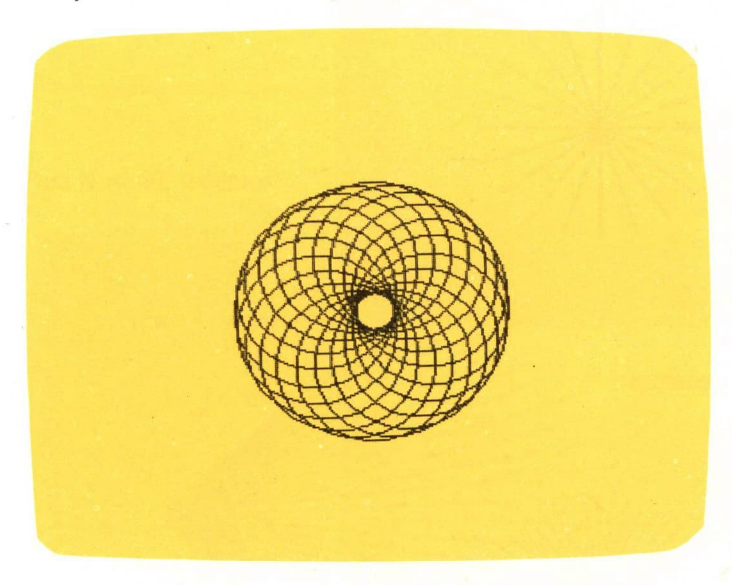
Flor para N = 16, R = 40 y RP = 47



Flor para N = 30, R = 25 y RP = 25



Flor para N = 20, R = 40 y RP = 30

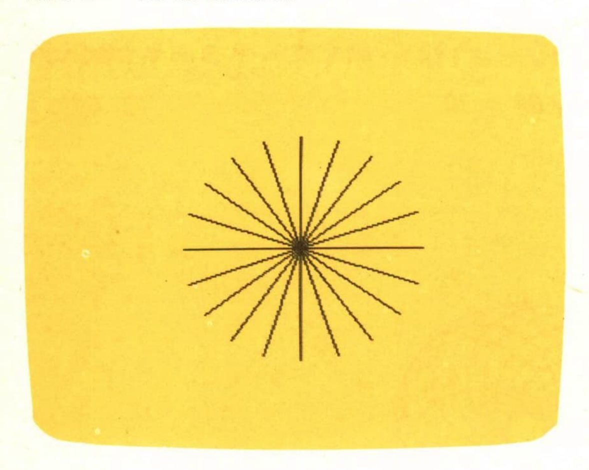


5. Rayos desde el centro de una circunferencia

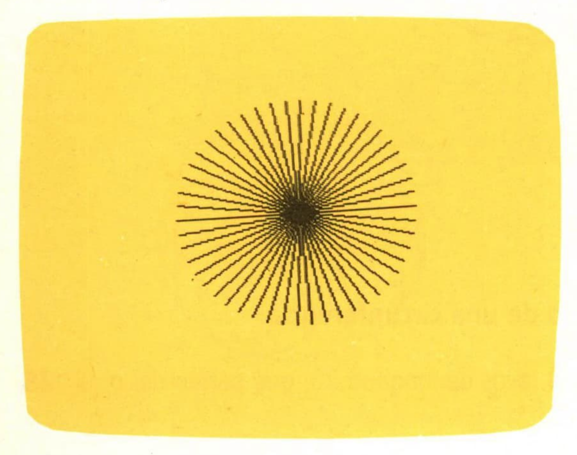
El programa siguiente traza N rayos de longitud 70, que parten del pixel 128, 88.

```
10  INPUT "NUMERO DE RAYOS"; N
20  FOR I = 1 TO N
30  PLOT 128, 88
40  LET X = 70 * COS (2 * PI/N * I)
50  LET Y = 70 * SIN (2 * PI/N * I)
60  DRAW X, Y
70  NEXT I
```

Para N = 20 se obtiene:



Para N = 50, tenemos:

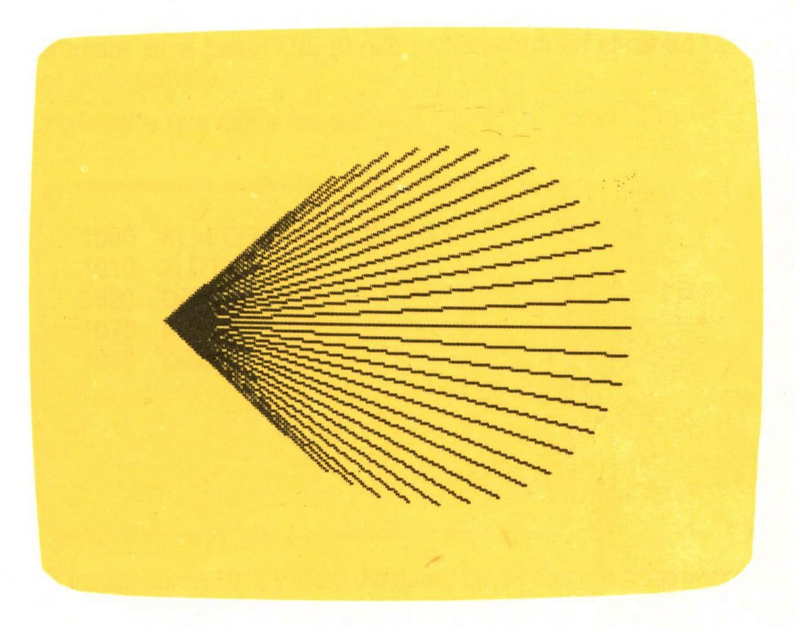


6. Rayos que parten de un punto situado a la izquierda de una circunferencia

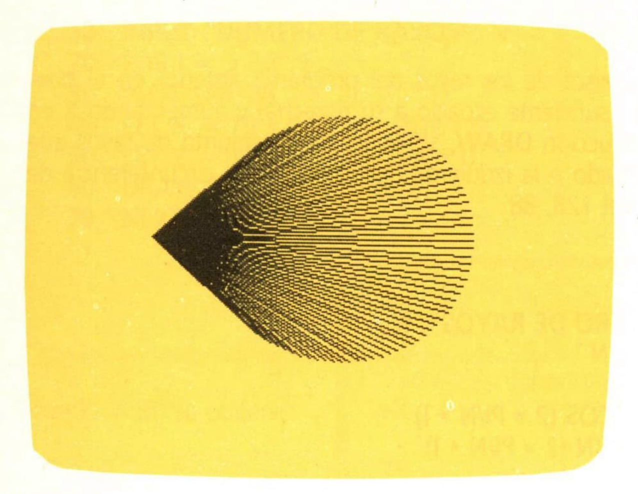
Localizando el centro emisor de los rayos del programa anterior en el pixel 28, 88 (para que quede suficiente espacio a su derecha) y aumentando X en 100 unidades en la instrucción **DRAW**, se obtiene un conjunto de rayos que parten de un punto situado a la izquierda del centro de la circunferencia de radio 70, y centro el pixel 128, 88.

```
10  INPUT "NUMERO DE RAYOS"; N
20  FOR I = 1 TO N
30  PLOT 28, 88
40  LET X = 70 * COS (2 * PI/N * I)
50  LET Y = 70 * SIN (2 * PI/N * I)
60  DRAW 100 + X, Y
70  NEXT I
```

Para N = 40, tenemos:



Para N = 100, obtenemos:



4. Representación gráfica de funciones

1. Planteamiento del problema

Para dibujar la gráfica de una función debemos tener en cuenta dos cuestiones previas:

— ¿Dónde situar el centro de los ejes?
— ¿Qué escala hay que elegir en cada eje?

Con el fin de contestar a estas preguntas estudiaremos una función sencilla: $F(x) = x^2$, y a partir de ella intentaremos sacar conclusiones generales.

2. Centrado de ejes

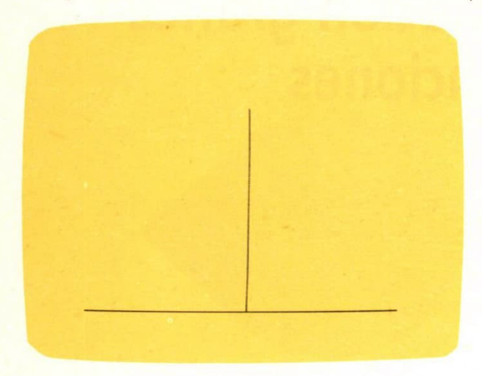
La gráfica de la función $F(x) = x^2$ es una parábola cuyo vértice es el punto (0, 0), y ocupa el primero y el segundo cuadrante. Luego los ejes deben estar centrados en el pixel (128, 0) que se encuentra en el centro de la línea inferior de la pantalla.

La subrutina que dibuja los ejes es:

```
1000 REM CENTRADO DE EJES
1010 PLOT 0, 0
1020 DRAW 255, 0
1030 PLOT 128, 0
1040 DRAW 0, 175
```

Las instrucciones 1010 y 1020 dibujan el eje de abscisas, y las instrucciones 1030 y 1040 dibujan el eje de ordenadas.

Al ejecutar esta subrutina se obtiene lo que muestra la pantalla.

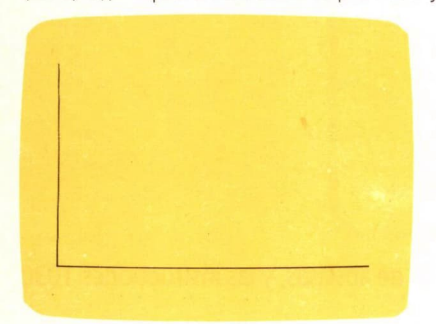


Si queremos que los ejes queden centrados en el pixel (C1, C2), basta generalizar la subrutina anterior cambiando 128 por C1 y 0 por C2 en las instrucciones **1030** y **1010**.

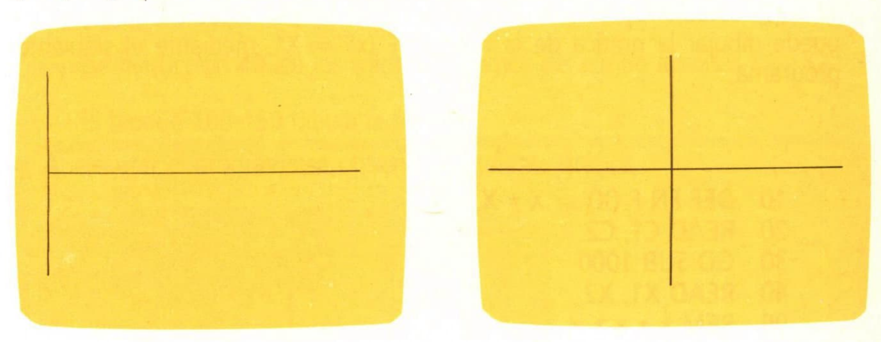
```
1000  REM CENTRADO DE EJES
1010  PLOT 0, C2
1020  DRAW 255, 0
1030  PLOT C1, 0
1040  DRAW 0, 175
```

Esta subrutina dibuja los ejes coordenados para cualquier posición del centro (C1, C2), pero aquí solamente consideraremos los siguientes centros:

(0, 0), que determina el primer cuadrante,
(128, 0), que determina el primero y el segundo cuadrante,

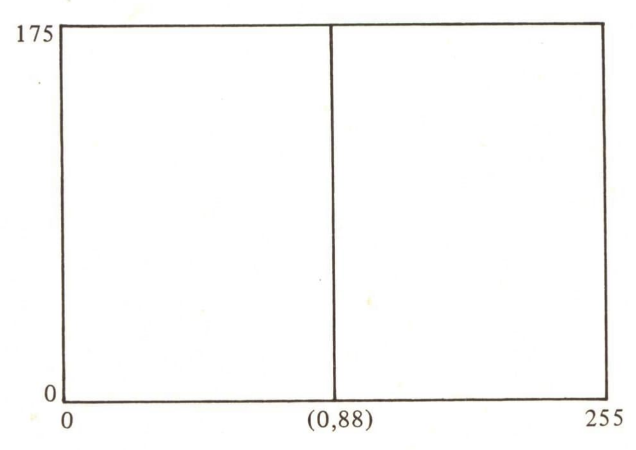


(0, 88), que determina el primero y el cuarto cuadrante y
(128, 88), que determina los cuatro cuadrantes.



3. Localización de la gráfica en los límites de la pantalla

- Si una vez elegido el centro de los ejes, se empieza a dibujar puntos de la función F (x) = x^2, puede ocurrir que algunos de ellos queden fuera de los límites de la pantalla. Ocurre, entonces, que la ejecución se detiene, apareciendo un mensaje de error. Es necesario, por tanto, evitar esta situación eligiendo adecuadamente los valores extremos de X a la izquierda y derecha de la pantalla, de tal manera que sus correspondientes valores F (x) no sobrepasen los valores 0 y 175 (límite inferior y superior de la pantalla).



Si el valor máximo que puede tomar F (x) es 175, ¿cuáles serán los valores extremos de X que hacen que F (x) no supere dicho valor?

Si $F(x) = x^2 = 175$ entonces $x = \pm \sqrt{175} = \pm 13.228757$

Luego los valores de X estarán comprendidos entre –13 y 13:

$$-13 < x < 13$$

- Los valores extremos de X a la izquierda y a la derecha del origen de coordenadas los indicaremos por X1 y X2, respectivamente. En el ejemplo que venimos estudiando, X1 = –13 y X2 = 13. Fijados estos extremos, ya se puede dibujar la gráfica de la función F (x) = X^2, mediante el siguiente programa:

```
  10 DEF FN F (X) = X * X
  20 READ C1, C2
  30 GO SUB 1000
  40 READ X1, X2
  99 REM * * * * *
 100 REM DIBUJO
 110 FOR X = X1 TO X2
 120 LET Y = FN F (X)
 130 PLOT C1 + X, C2 + Y
 140 NEXT X
 150 STOP
 999 REM * * * * * * * * * * *
1000 REM CENTRADO DE EJES
1010 PLOT 0, C2
1020 DRAW 255, 0
1030 PLOT C1, 0
1040 DRAW 0, 175

1070 RETURN
8999 REM * * * *
9000 REM DATOS
9010 DATA 128,0
9020 DATA –13,13
```

Este programa funciona así:

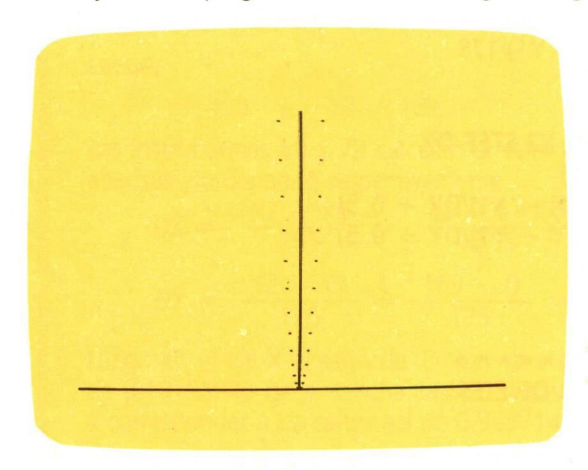
— La instrucción **10** define la función FN F (X) = X * X (No se pone X ↑ 2, pues el ZX Spectrum no admite potencias de números negativos.)

— La instrucción **20** lee los valores del centro C1 y C2, y la instrucción **30** llama a la subrutina de "centrado de ejes" para que dibuje los ejes de coordenadas centrados en C1, C2.

— La instrucción **40** lee los valores extremos del eje de abscisas X1 y X2.

— El bloque **100-150** dibuja la función.

Al ejecutar el programa se obtiene la siguiente gráfica.



Se observa que la gráfica obtenida está formada por puntos bastante distantes, y que se ha desperdiciado gran parte de la pantalla, a izquierda y a derecha del eje de ordenadas. Esto se puede evitar eligiendo adecuadamente las escalas en ambos ejes.

4. Elección de escalas en los ejes

- Para determinar la escala en cada eje elegiremos las siguientes variables:

X1: extremo inferior del eje de abscisas

X2: extremo superior del eje de abscisas

Y1: extremo inferior del eje de ordenadas

Y2: extremo superior del eje de ordenadas

DX: valor de la división en el eje X

DY: valor de la división en el eje Y

X3: abscisa del punto de la pantalla que se va a imprimir

Y3: ordenada del punto de la pantalla que se va a imprimir

El programa que incluye la utilización de escalas es el siguiente:

```
10 DEF FN F (X) = X * X
20 READ C1, C2
30 GO SUB 1000
40 READ X1, X2
50 READ Y1, Y2
60 LET DX = (X2 - X1)/255
70 LET DY = (Y2 - Y1)/175
99 REM * * * * *
100 REM DIBUJO
110 FOR X = X1 TO X2 STEP DX
120 LET Y = FN F(X)
130 LET X3 = INT ((X - X1)/DX + 0. 5)
140 LET Y3 = INT ((Y - Y1)/DY + 0. 5)
180 PLOT X3, Y3
190 NEXT X
200 STOP
999 REM * * * * * * * * * * *
1000 REM CENTRADO DE EJES
1010 PLOT 0, C2
1020 DRAW 255, 0
1030 PLOT C1, 0
1040 DRAW 0, 175
1070 RETURN
8999 REM * * * *
9000 REM DATOS
9010 DATA 128, 0
9020 DATA -13, 13
9030 DATA 0, 169
```

Veamos cómo funciona este programa:

- La instrucción **10** define la función F (X) = X * X.
- La instrucción **20** lee las coordenadas del centro (C1, C2).
- La **30** llama a la subrutina que dibuja los ejes.
- La **40** lee los valores extremos del eje de abscisas y la **50** los valores extremos del eje de ordenadas.

Como se ha visto, estos valores extremos no son arbitrarios sino que se calculan teniendo en cuenta los límites de la pantalla.

En este ejemplo, X1 y X2 se determinaron anteriormente:

$$X1 = -13 \quad y \quad X2 = 13$$

Los extremos de Y se calculan considerando que

$$F(-13) = (-13)^2 = 169 \quad y \quad F(13) = 13^2 = 169$$

y que se cumple $F(X) = X^2 \geqslant 0$

Luego,

$$Y1 = 0 \quad e \quad Y2 = 169$$

— Las instrucciones **60** y **70** calculan el valor de la división en los ejes de abscisas y ordenadas, respectivamente.

$$DX = \frac{X2 - X1}{255} = \frac{13 - (-13)}{255} = \frac{26}{255} = 0.10196078$$

$$DY = \frac{Y2 - Y1}{175} = \frac{169 - 0}{175} = 0.96571429$$

Luego en el eje X el salto de 1 pixel va a corresponder a un salto real de 0.10196078 (se ensancha casi 10 veces la gráfica), y en el eje Y va a corresponder a un salto real de 0.96571429.

— La línea **110 FOR X = X1 TO X2 STEP DX** abre el bucle que dibuja la función. En total se van a imprimir 255 pixels, pues X varía desde X1 = = −13 hasta X2 = 13 pero con paso DX = 0.10196078.

— La instrucción **120 LET Y = FN F (X)** calcula los valores de la función $F(X) = X^2$ correspondientes a los valores comprendidos entre X1 y X2.

— Las instrucciones **130** y **140** calculan los valores X3 e Y3 en función de X e Y, respectivamente. Aquí se tienen en cuenta los extremos X1, X2 y las escalas elegidas DX y DY en los ejes coordenados.

Los valores de la variable X3 estarán comprendidos entre 0 y 255, y se calculan mediante la expresión:

$$X3 = INT\left(\frac{X - X1}{DX} + 0.5\right)$$

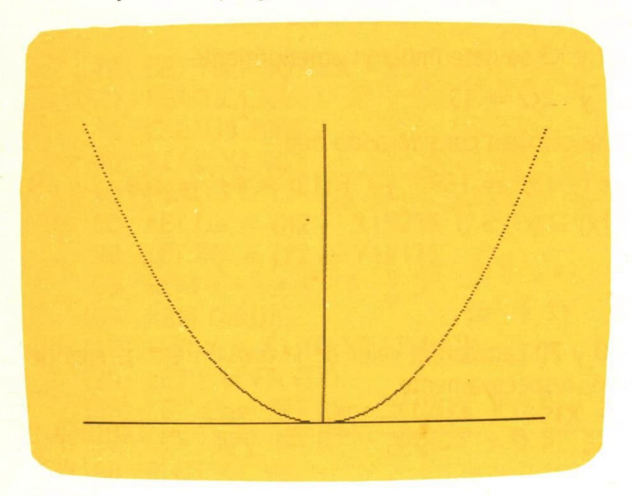
Los valores de Y3 estarán comprendidos entre 0 y 169, y se calculan mediante la expresión:

$$Y3 = INT\left(\frac{Y - Y1}{DY} + 0.5\right)$$

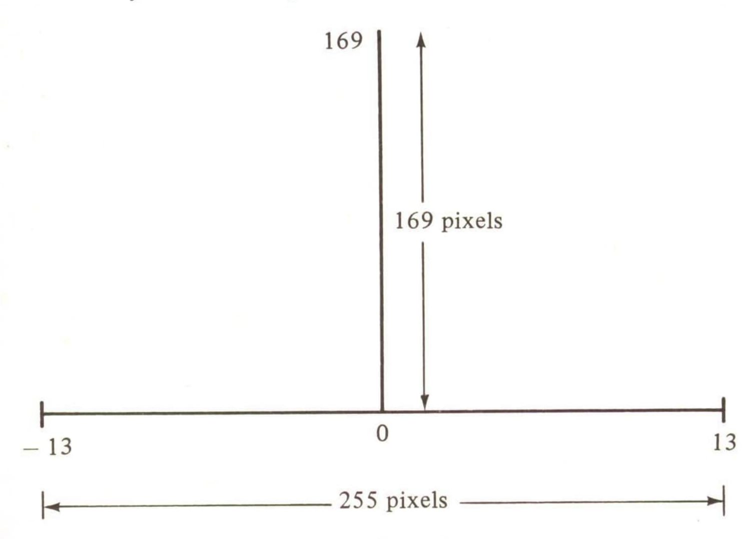
— La instrucción **180 PLOT X3, Y3** imprime el pixel correspondiente.

— La línea **190 NEXT X** obliga repetir el bucle y la **200 STOP** para el proceso.

Si se ejecuta el programa se obtiene lo que muestra la pantalla:



Ahora se aprecia mejor la gráfica de la función F (X) = X^2, aunque tiene el inconveniente de estar bastante deformada por haber elegido escalas diferentes en el eje de abscisas y en el de ordenadas.



La gráfica está ensanchada considerablemente, y este ensanchamiento viene dado por la relación entre ambas escalas DX y DY. Dicha relación es de 0.10196078 a 0.96571429:

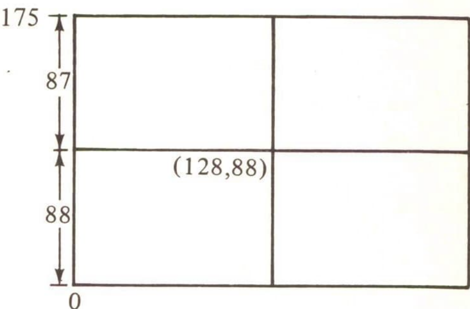
0.10196078 / 0.96571429 = 0.1055806

que impone un ensanchamiento de casi 10 veces.

5. Gráficas de algunas funciones

- **$F(X) = X^3$**

Como F (X) puede tomar valores negativos y positivos interesa centrar los ejes en el pixel (128, 88).



Si hacemos $F(X) = X^3 = 87$, se obtiene $X = \sqrt{87} = 4.4310476$

Elegiremos, entonces:

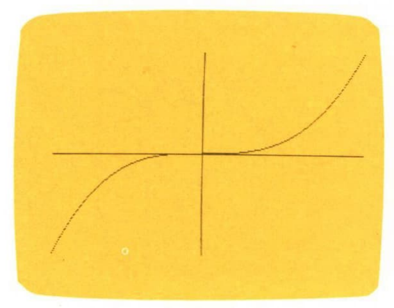
$X1 = -4$ $\quad\quad$ $X2 = 4$

$Y1 = -64$ $\quad\quad$ $Y2 = 64$

En el programa general hay que cambiar las instrucciones siguientes:

```
  10 DEF FN F (X) = X * X * X
9010 DATA 128, 88
9020 DATA -4, 4
9030 DATA -64, 64
```

Al ejecutar el programa se obtiene la siguiente pantalla:



- **F (X) = sen X**

  La función F (X) = sen X es periódica, de período 2 π. Luego X variará entre X1 = 0 y X2 = 2 π . Como además, el seno varía entre −1 y 1,

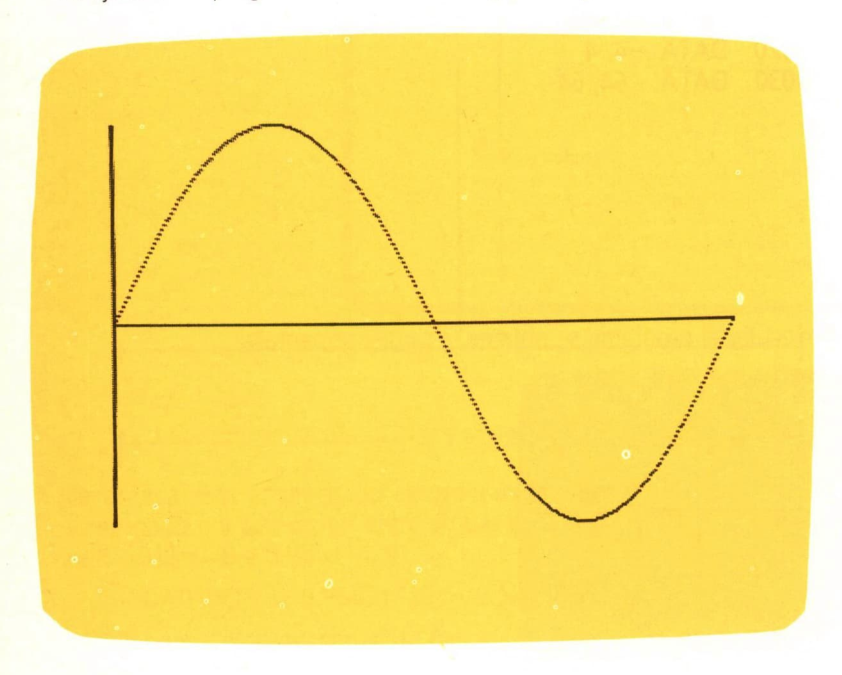
  $$-1 \leqslant \text{sen } X \leqslant 1$$

  los valores extremos de Y serán Y1 = −1 e Y2 = 1. Es evidente que el centro habrá que localizarlo en el pixel (0, 88).

  Las instrucciones que hay que cambiar en el programa general son éstas:

```
  10 DEF FNX (X) = SIN X
9010 DATA 0, 88
9020 DATA 0,2 * PI
9030 DATA -1, 1
```

Al ejecutar el programa se obtiene la siguiente gráfica:



- **F (X) = cos X**

La función F (X) = cos X también es periódica, de período 2 π, luego X variará entre X1 = 0 y X2 = 2 π. Como además se cumple $-1 \leqslant$ sen X $\leqslant 1$, los valores extremos de Y serán los siguientes

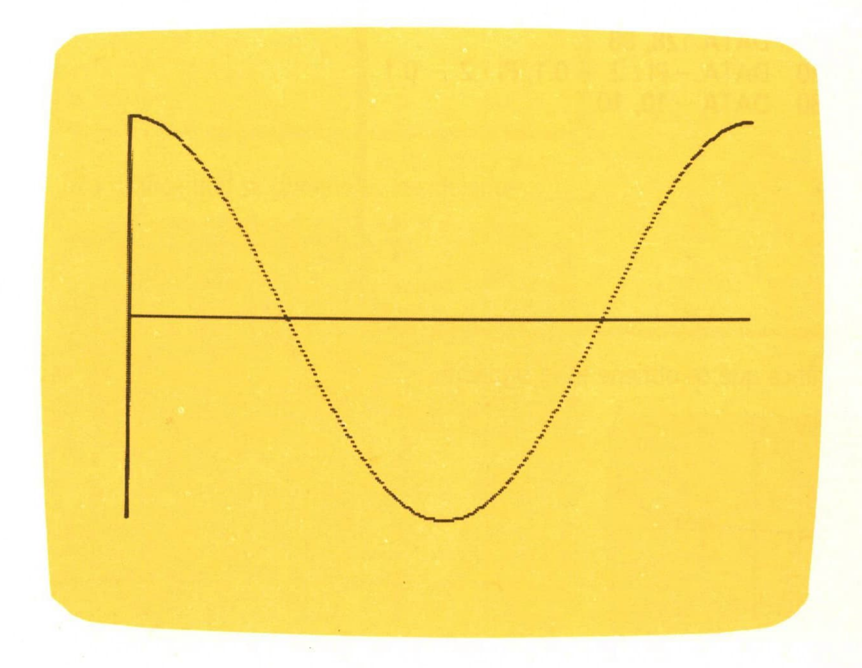
Y1 = −1 e Y2 = 1

El centro se situará en el pixel (0, 88).

Las instrucciones que hay que cambiar en el programa general son éstas:

```
10   DEF FNF (X) = COS X
9010 DATA 0, 88
9020 DATA 0, 2 * PI
9030 DATA -1, 1
```

Al ejecutar el programa se obtiene la siguiente gráfica:



- **F (X) = TAN X**

La función F (X) = TAN X también es periódica, pero de período π y como es discontinua en

$$-\frac{\pi}{2}, \frac{\pi}{2}$$

elegiremos como valores extremos de X los siguientes:

$$X1 = -\frac{\pi}{2} + 0.1$$

$$X2 = \frac{\pi}{2} - 0.1$$

Y como.

$$F(X1) = TAN\left(-\frac{\pi}{2} + 0.1\right) = -9.9666444$$

$$F(X2) = TAN\left(-\frac{\pi}{2} - 0.1\right) = 9.9666444$$

hacemos

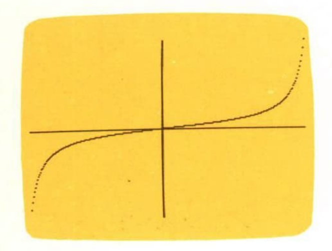
Y1 = −10
Y2 = 10

El centro estará en el pixel (128, 88).

Las instrucciones que hay que cambiar en el programa general son éstas:

```
  10 DEF FN F (X) = TAN X
9010 DATA 128, 88
9020 DATA -PI / 2 + 0.1, PI / 2 - 0.1
9030 DATA -10, 10
```

La gráfica que se obtiene es la siguiente:



- **F (X) = $\sqrt{X}$**

La función F (X) = $\sqrt{X}$ da el valor positivo de la ***raíz cuadrada de X,*** y solamente definida para $X \geq 0$. Se tiene entonces que si

$X1 = 0$
$X2 = 255$

la raíz cuadrada de estos valores es

$\sqrt{255} = 15.968719$

Luego los valores extremos de Y son:

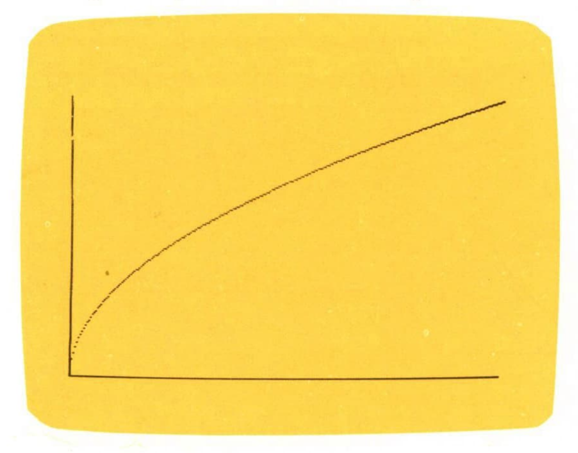
$Y1 = 0$
$Y2 = 15.97$

El centro estará en el pixel (0, 0).

Las instrucciones que hay que cambiar en el programa general son:

```
  10  DEF FN F (X) = SQR X
9010  DATA 0, 0
9020  DATA 0, 255
9030  DATA 0, 15.97
```

La gráfica que se obtiene es la siguiente:



- **F (X) = e^x**

La función exponencial F (X) = e^x siempre es positiva y al ser $e^0 = 1$, en el eje de ordenadas debemos tomar un valor pequeño para que se pueda apreciar la gráfica de la función para valores de X negativos.

Si elegimos como valor extremo superior de Y

Y2 = 20

el correspondiente valor extremo de X a la derecha se obtiene así

$$e^x = 20 \Rightarrow \ln e^x = \ln 20 \Rightarrow x = 2.9957323$$

Esto nos lleva a elegir como valores extremos de X los siguientes:

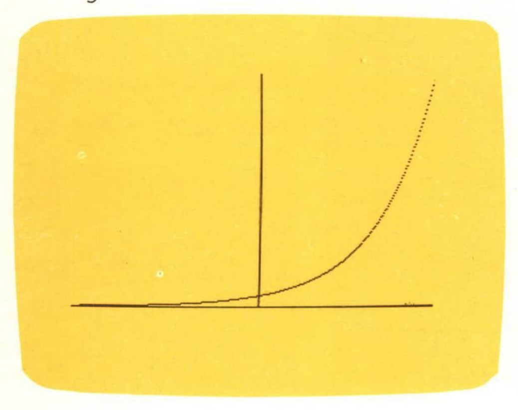
X1 = −3
X2 = 3

El centro será el pixel (128, 0).

Las instrucciones que hay que cambiar en el programa general son:

```
  10 DEF FN F (X) = EXP X
9010 DATA 128, 0
9020 DATA -3, 3
9030 DATA 0, 20
```

El resultado que se obtiene al ejecutar el programa es el que muestra la fotografía:



- **F (X) = ln X**

La función logaritmo natural o neperiano F (X) = ln X sólo está definida para valores de X positivos y al ser ln 1 = 0 el valor máximo de X no debe ser muy grande para que se pueda apreciar la parte negativa de la función.

Por ejemplo, si elegimos como valores extremos de X

X1 = 0.1
X2 = 20

los correspondientes valores extremos de Y se obtienen así:

ln 0.1 = −2.3025851
ln 20 = 2.9957323

Elegimos como valores extremos de Y los siguientes

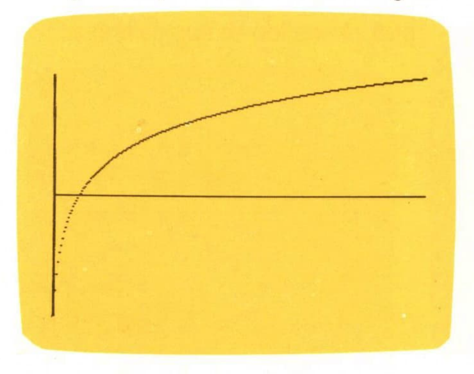
Y1 = −2.3
Y2 = 3

El centro será el pixel (0, 88).

Las instrucciones que hay que cambiar en el programa general son:

```
  10  DEF FN F (X) = LN X
9010  DATA 0, 88
9020  DATA 0.1, 20
9030  DATA −2.3, 3
```

La gráfica que se obtiene es la siguiente:



- **$F(X) = \frac{1}{X}$**

La función $F(X) = \frac{1}{X}$ es discontinua para $X = 0$, luego hay que evitar valores próximos al cero; además X no debe ser ni muy pequeña ni muy grande, pues de lo contrario $F(X) = \frac{1}{X}$ se haría demasiado grande o demasiado pequeña, respectivamente. Así se obtendrá una buena representación.

Teniendo en cuenta las anteriores consideraciones, tomamos:

$X1 = -5 \qquad X2 = 5$

$Y1 = -5 \qquad Y2 = 5$

y como centro, el pixel (128, 88), pues X e Y van a ser ambos positivos.

Al dispararse la función para valores de X muy próximos al cero (discontinuidad en el cero) hay que añadir una instrucción como la siguiente:

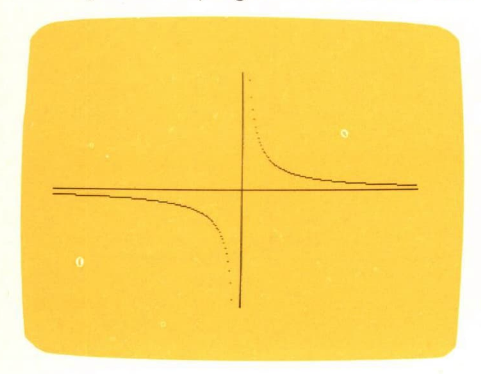
115 IF −0.2 < = X AND X < = 0.2 THEN GOTO 190

que evita la división por cero.

Las instrucciones que hay que cambiar o añadir al programa general son:

```
  10  DEF FN F (X) = 1/X
 115  IF −0.2 < = X AND X <
= 0.2 THEN GO TO 190
9010  DATA 128, 88
9020  DATA −5, 5
9030  DATA −5, 5
```

Al ejecutar el programa se obtiene esta gráfica:



- **$F(X) = \frac{\text{sen } X}{X}$**

La función $F(X) = \frac{\text{sen } X}{X}$

es continua, pues cuando X se aproxima a cero (0) se tiene que $F(X) = \frac{\text{sen } X}{X}$

se aproxima a 1. $\frac{\text{sen } 0.1}{0.1} \simeq 0.99833417$

y cuando x se hace grande (en valor absoluto), $\frac{\text{sen } X}{X}$ tiende a cero,

pues el sen X se mantiene entre -1 y 1. Así: $\frac{\text{sen } 15}{15} = 0.043352523$

Teniendo en cuenta estas propiedades, elegimos como valores extremos, por ejemplo, los siguientes:

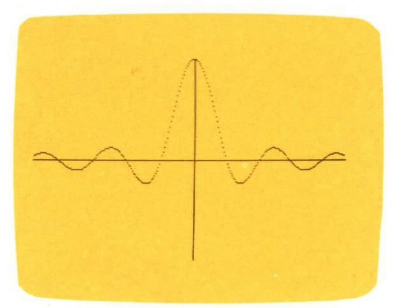
$X1 = -15$ $\quad$ $X2 = 15$
$Y1 = -1$ $\quad$ $Y2 = 1$

El centro será el pixel (128, 88), pues X e Y pueden ser positivos y negativos.

Las instrucciones que hay que cambiar en el programa general son:

```
  10 DEF FN F (X) = SIN X/X
9010 DATA 128, 88
9020 DATA -15, 15
9030 DATA -1, 1
```

La gráfica que se obtiene es ésta:



- **F (X) =** $\frac{\cos X}{X}$

Esta función es discontinua para X igual a cero, luego debemos añadir al programa general una instrucción como ésta:

115 IF −1 < = X AND X < = 1 THEN GOTO 190

dado que si X tomara un valor entre −1 y 1 la función se saldría de la pantalla.

Además, si X se hace grande (en valor absoluto) $\frac{\cos X}{X}$ tiende a cero, al mantenerse cos X entre −1 y 1. Así: $\frac{\cos 15}{15} = -0.050645861$

Luego tomaremos por ejemplo los siguientes valores extremos

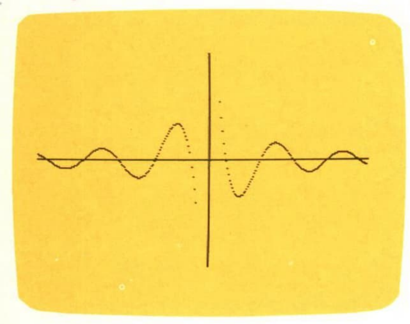
X1 = −15 X2 = 15
Y1 = −1 Y2 = 1

Como los valores de X e Y pueden ser positivos y negativos, el centro estará en el pixel (128, 88).

Las instrucciones que hay que añadir o cambiar en el programa general son:

```
  10  DEF FN F (X) = COS X/X
 115  IF −1 < = X AND X < =
1 THEN GO TO 190
9010  DATA 128, 88
9020  DATA −15, 15
9030  DATA −1, 1
```

La gráfica que se obtiene es la que muestra la fotografía:



- **F (X) = sen X + cos X**

  La función F (X) = sen X + cos X es periódica, de período 2 π, y al ser la suma del seno y del coseno, F (X) variará entre −1 −1 = −2 y 1 + 1 = 2.

  Los valores extremos que debemos tomar son:

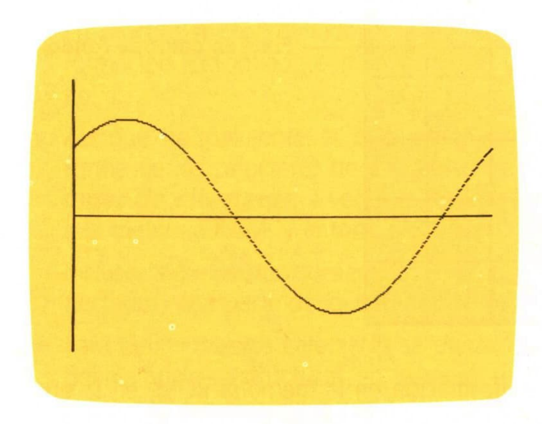
  $X1 = 0 \qquad X2 = 2\pi$
  $Y1 = -2 \qquad Y2 = 2$

  Como X sólo toma valores positivos el centro lo tomamos en el pixel (0, 88).

  Las instrucciones que hay que cambiar en el programa general son:

```
  10 DEF FN F (X) = SIN X + COS X
9010 DATA 0, 88
9020 DATA 0, 2 * PI
9030 DATA −2, 2
```

La gráfica que se obtiene es la siguiente:

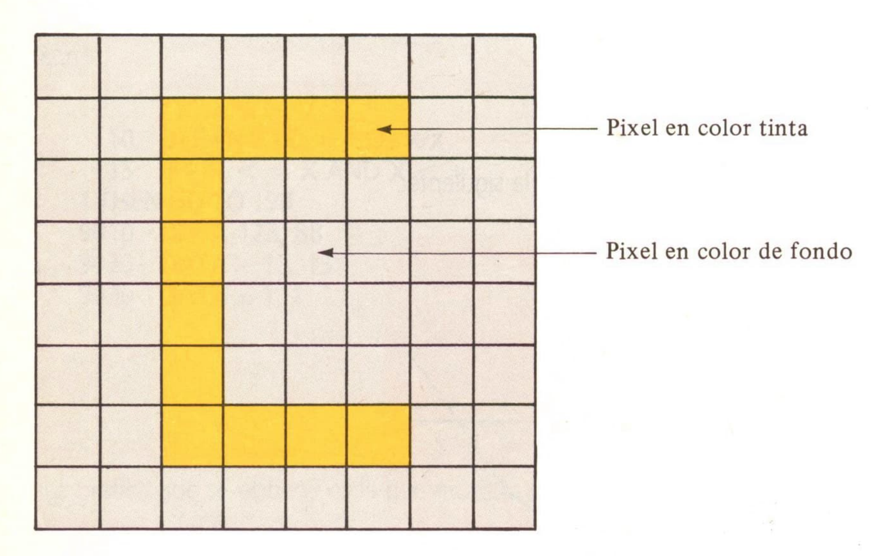


5. Creación de caracteres y movimientos

1. Cómo dibuja el microordenador caracteres en la pantalla

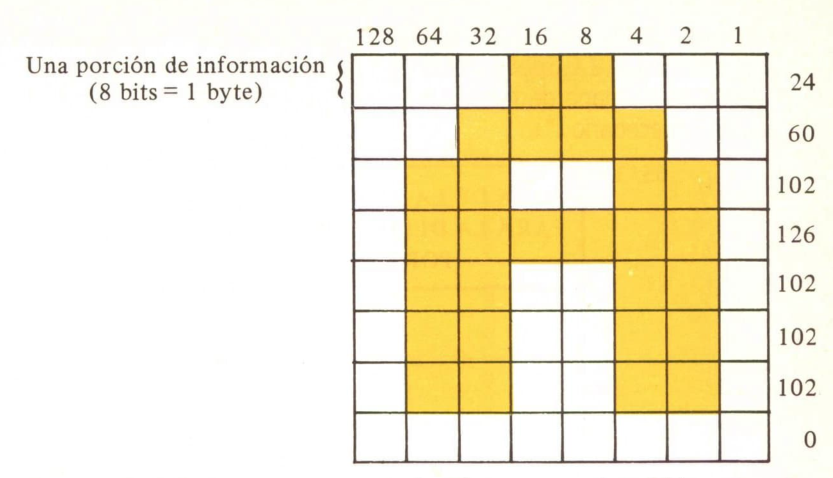
- Si observamos detenidamente un carácter en la pantalla, por ejemplo, la letra A, veremos que está compuesto por un conjunto de pequeños cuadritos, llamados ***pixels.***

Para que el microordenador dibuje un carácter en la pantalla debe recibir la información necesaria para dibujar en ***color*** tinta los pixels que definen el carácter, y los que se encuentran dentro del área del mismo, en el ***color de fondo*** (papel), tal como se indica en la siguiente figura, que representa los pixels muy aumentados.



El microordenador busca esta información en la memoria ROM, en la que están almacenadas ***ocho posiciones de información*** por cada carácter. Cada una de estas posiciones, compuestas de 8 bits (un byte), define una línea horizontal del carácter.

Podemos, entonces, obtener un carácter a partir de una red de 8 × 8 cuadritos, ennegreciendo los que corresponden a los pixels en color de tinta de un carácter determinado.



Un carácter queda definido por una sucesión de 8 bytes, o sea, 8 × 8 bits, correspondientes a 64 pixels.

En esta figura, que representa la letra A (muy aumentada), los números que se encuentran encima de cada columna corresponden al valor de cada bit, expresado en base decimal; y los números situados del lado derecho son el resultado de sumar los valores de los bits conectados, en cada fila.

2. Cómo modificar un carácter

- En muchos microordenadores la parte de la memoria que almacena la información relacionada con la forma de los caracteres es accesible al usuario.

  Aunque normalmente el microordenador recibe la información sobre la forma de los caracteres de una parte de la memoria ROM, llamada ***generador de caracteres,*** a veces es posible trasladar toda o una parte de ésta a la memoria RAM, y entonces el usuario puede modificarla a voluntad.

  Incluso, algunos microordenadores almacenan automáticamente, al ser conectados, una parte del generador de caracteres en la memoria RAM.

- Para definir nuevos caracteres se utiliza la instrucción POKE, la cual puede definir nuevas estructuras en la memoria RAM mediante números, expresados en base decimal a partir de una cierta dirección de memoria.

  A este respecto es necesario señalar que durante el proceso de definición de un nuevo carácter el primitivo se pierde y el definido por el usuario ocupa su zona de memoria con las características específicas que el usuario quiso adjudicarle. La forma del nuevo carácter se puede observar haciéndole aparecer en pantalla mediante instrucciones PRINT.

Veamos cómo se lleva a cabo el proceso de definición de un carácter en el microordenador ZX Spectrum. Éste copia automáticamente en la parte superior de su mapa de memoria RAM las estructuras de las letras mayúsculas del abecedario (*).

65536

ZONA DE LA MEMORIA RESERVADA PARA LA DEFINICIÓN DE GRÁFICOS POR EL USUARIO.

Previamente es necesario conocer la posición de la memoria (su dirección), en la que se inicializará el almacenamiento del primer carácter definido por el usuario. Esto se consigue mediante la instrucción USR. Por ejemplo, USR "A" da la dirección de la memoria correspondiente al primer byte de los ocho que definen el carácter A. Así, el programa

```
10  FOR I = 0 TO 7
20  PRINT PEEK USR "A" + I
30  NEXT I
```

proporciona ocho números, expresados en base decimal, correspondientes a las direcciones de los ocho bytes que definen la letra A.

En otros microordenadores hay que consultar en el manual la dirección en la que comienza la zona de la memoria reservada a la definición de caracteres por el usuario.

(*) Estos caracteres aparecen en la pantalla del ZX Spectrum cuando se coloca en modo gráfico (en este modo aparece una G en el cursor).

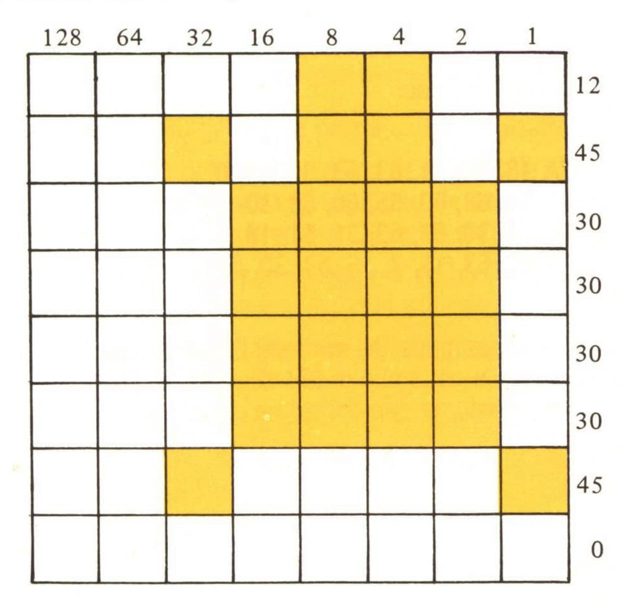
Una cucaracha móvil

Si en lugar de una A deseamos almacenar en su propia zona una pequeña cucaracha, escribiremos este conjunto de instrucciones:

```
 5 LET M = USR "A"
10 FOR I = 0 TO 7
20 READ B
30 POKE M + I, B
40 NEXT I
50 DATA 12, 45, 30, 30, 30, 30, 45, 0
```

Una vez pulsadas **RUN** y **ENTER** la instrucción **POKE** va colocando en las posiciones de la memoria identificadas por las direcciones M los valores correspondientes a los diversos valores que producen el cambio de carácter. Estos valores están guardados en la instrucción DATA.

La figura almacenada es la siguiente:



Esta figura se puede visualizar en la pantalla escribiendo **PRINT A,** estando el microordenador ZX Spectrum en modo gráfico.

Si añadimos el siguiente conjunto de instrucciones conseguiremos que esta cucaracha se desplace por la pantalla:

```
 60 FOR X = 0 TO 21
 70 FOR Y = 21 TO 1 STEP -1
 80 PRINT AT Y, X; "A"
 90 PAUSE 10
100 PRINT AT Y, X, " "
110 NEXT Y
120 NEXT X
```

Dibujo de un muñeco

Cambiando los valores de la instrucción **DATA** se pueden definir otras figuras. Así, poniendo

50 DATA 28, 28, 28, 8, 62, 8, 20, 20, 34

se obtiene un muñeco pequeño.

Dibujo de diferentes tipos de invasores

En forma análoga, poniendo

```
50 DATA 18, 51, 51, 63, 63, 30, 12, 0
50 DATA 30, 63, 60, 56, 60, 63, 30, 0
50 DATA 12, 30, 63, 63, 51, 51, 18, 0
50 DATA 30, 63, 15, 7, 15, 63, 30, 0
```

se obtienen diferentes tipos de invasores, apuntando cada uno para un lado.

3. Una pelota rebotando

- El programa siguiente define en la letra A una pelota, que mediante las instrucciones siguientes se hará rebotar en los cuatro bordes de la pantalla.

```
  5 RANDOMIZE
 10 LET M = USR "A"
 20 FOR I = 0 TO 7
 30 READ B
 40 POKE M + I, B
 50 NEXT I
 60 DATA 30, 63, 63, 63, 63, 30, 0
 70 LET XB = 2 : LET XH = 32
 80 LET YB = 2 : LET YH = 22
 90 LET XP = XB + INT (RND * 6) + 1 : LET YP = YB + INT
(RND * 6) + 1
100 LET VX = 1 : LET VY = 1
110 IF XP < XB + 1 OR XP > XH - 1 THEN LET VX = -VX :
GO TO 160
120 IF YP < YB + 1 OR YP > YH - 1 THEN LET VY = -VY :
GO TO 160
130 PRINT AT YP, XP; "A"
140 PAUSE 10
150 PRINT AT YP, XP; " "
160 LET XP = XP + VX : LET YP = YP + VY
170 GO TO 110
```

Las instrucciones **10** a **60** definen la pelota, colocándola en la letra A.

Las instrucciones **70-80** definen los límites de la pantalla; la instrucción **90** establece, en combinación con la **5**, un punto de partida XP, YP aleatorio, y la **100** el valor con que incrementarán XP e YP en la instrucción **160.**

Las instrucciones **110** y **120** comprueban si se ha llegado al borde de la pantalla; si esto ocurre cambian de signo el valor de VX y VY.

Finalmente, la instrucción **130** imprime en la pantalla la pelota, la **140** establece un tiempo de espera y la **150** imprime, en el lugar en el que se encontraba la pelota, un espacio en blanco que simulará el movimiento.

4. Manejo de la instrucción inkey$. Pulgarcito

- El programa siguiente, además de definir un pequeño hombrecito en la letra A, permite desplazarlo por medio del teclado e incluso consigue que deje un rastro de puntos tras él.

Se utiliza para esto la instrucción **INKEY$**, que permite la lectura del teclado para analizar qué tecla se está pulsando en un momento determinado, sin que el programa por ello se detenga.

El carácter correspondiente a la tecla pulsada puede almacenarse en una variable de cadena.

```
 10 LET M = USR "A"
 20 FOR I = 0 TO 7
 30 READ Q
 40 POKE M + I, Q
 50 NEXT I
 60 DATA 28, 28, 28, 8, 62, 8, 20, 34
100 LET X = 10 : LET Y = 10
110 LET C$ = INKEY$ : IF C$ < > " " THEN GOTO 160
120 PRINT AT Y, X; "A"
130 PAUSE 10
140 PRINT AT Y, X; " "
150 GO TO 110
160 IF C$ = "5" AND X > 0 THEN LET X = X - 1
170 IF C$ = "8" AND X < 31 THEN LET X = X + 1
180 IF C$ = "6" AND Y < 21 THEN LET Y = Y + 1
190 IF C$ = "7" AND Y > 0 THEN LET Y = Y - 1
200 GO TO 110
```

Las instrucciones **10** a **60** definen en la letra A un hombrecito, que se coloca en el punto X = 10, Y = 10.

La instrucción **110** lee el teclado y almacena el carácter leído en C$.

Las instrucciones **120** a **130** imprimen el carácter en movimiento.

Las instrucciones **160** a **190** varían las coordenadas para realizar los diferentes desplazamientos (*) y comprueban si se ha llegado a algún borde de la pantalla. En este caso no permiten el desplazamiento.

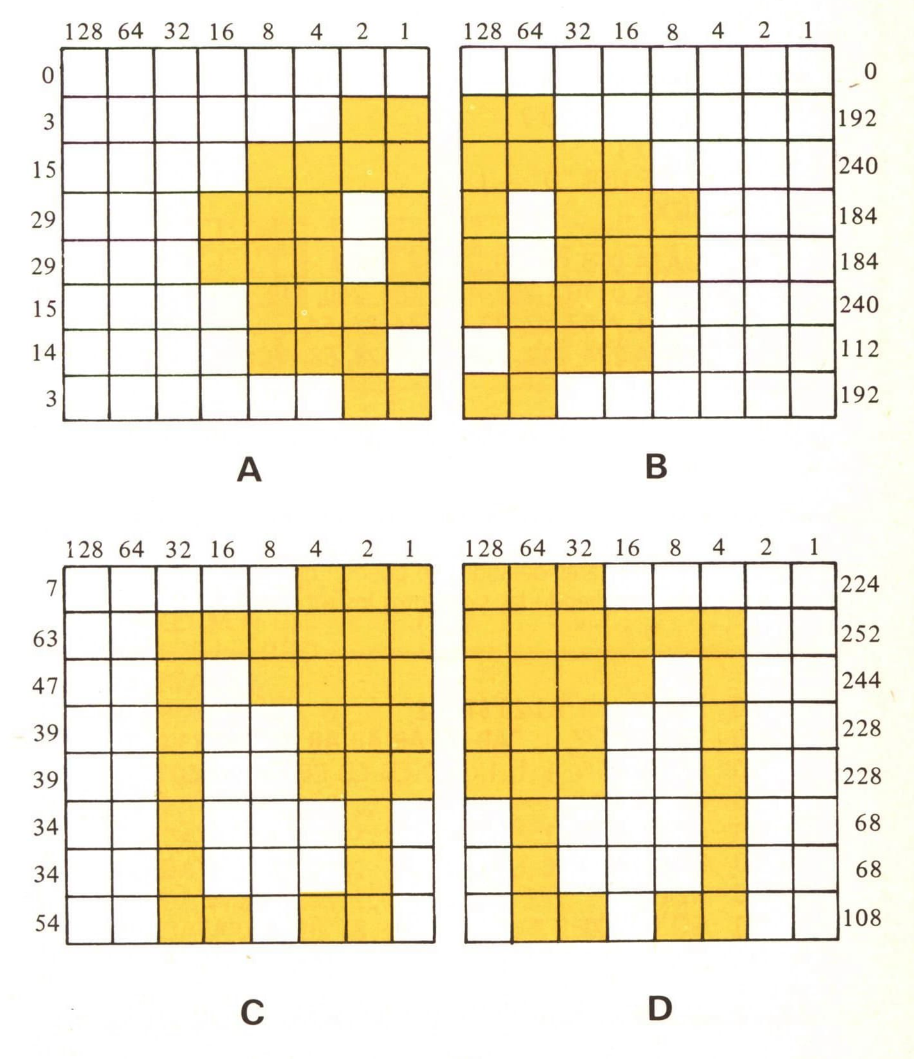
(*) En el ZX Spectrum en las teclas correspondientes al 5, 8, 6 y 7 están las flechas que indican los sentidos de desplazamiento del cursor.

5. Combinación de diferentes caracteres definidos

- Combinando varios caracteres definidos se pueden obtener figuras de mayor tamaño que los simples caracteres individuales.

Construcción de un invasor extraterrestre

El programa siguiente define cuatro caracteres que, combinados, producen un personaje extraterrestre de mayor tamaño, con las características ilustradas en esta figura.



```
A   10 FOR I = 0 TO 7
    20 READ X
    30 POKE USR "A" + I, X
    40 NEXT I

B   50 FOR I = 0 TO 7
    60 READ Y
    70 POKE USR "B" + I, Y
    80 NEXT I

C   90 FOR I = 0 TO 7
   100 READ Z
   110 POKE USR "C" + I, Z
   120 NEXT I

D  130 FOR I = 0 TO 7
   140 READ T
   150 POKE USR "D" + I, T
   160 NEXT I

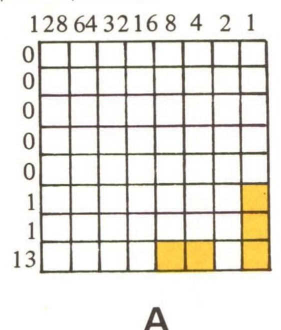
   500 DATA 0, 3, 15, 29, 29, 15, 14, 3
   510 DATA 0, 192, 240, 184, 184, 240, 112, 192
   520 DATA 7, 63, 47, 39, 39, 34, 34, 54
   530 DATA 224, 212, 244, 228, 228, 68, 68, 108
```

Estos cuatro nuevos caracteres definidos se pueden mover vertical y conjuntamente en la pantalla, añadiendo las siguientes instrucciones:

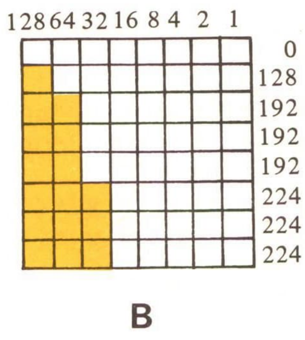
```
700 FOR Y = 1 TO 20 STEP 2
710 PRINT AT Y, 1; "AB AB AB AB AB AB AB"
720 PRINT AT Y + 1, 1; "CD CD CD CD CD CD CD"
730 PAUSE 20
740 PRINT AT Y, 1; "  "
750 PRINT AT Y + 1, 1; "  "
760 NEXT Y
770 GO TO 700
```

Barcos, aviones y cohetes

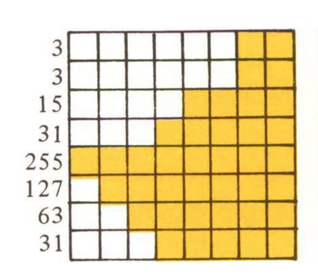
- Mediante un programa más breve que el anterior se puede conseguir que el siguiente barco, formado por cinco caracteres definidos, se desplace también por la pantalla.



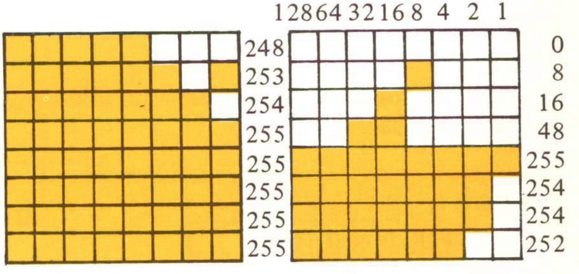
A



B



C



D

E

```
 10 FOR P = 0 TO 4
 20 LET M = USR "A" + P * 8
 30 FOR I = 0 TO 7
 40 READ Q
 50 POKE M + I, Q
 60 NEXT I
 70 NEXT P
500 DATA 0, 0, 0, 0, 0, 1, 1, 13
510 DATA 0, 128, 192, 192, 192, 224, 224, 224
520 DATA 3, 3, 15, 31, 255, 127, 63, 31
530 DATA 248, 253, 254, 255, 255, 255, 255, 255
540 DATA 0, 8, 16, 48, 255, 254, 254, 252
```

Este conjunto de instrucciones va modificando los cinco caracteres A, B, C, D y E, según se ejecuta el bucle **10-70**, desde P = 0 a 4, y que están almacenados en la memoria en posiciones consecutivas.

Si, además, añadimos las instrucciones

```
 80 FOR X = 0 TO 28
 90 PRINT AT 10, X; "AB"
100 PRINT AT 11, X; "DEF"
110 PAUSE 10
120 PRINT AT 10, X; "  "
130 PRINT AT 11, X; "   "
```

conseguiremos que el barco se desplace horizontalmente por la pantalla.

- En forma análoga podemos definir un avión, y desplazarlo horizontalmente por la pantalla

```
 10 FOR P = 0 TO 2
 20 LET M = USR "A" + P * 8
 30 FOR I = 0 TO 7
 40 READ Q
 50 POKE M + I, Q
 60 NEXT I
 70 NEXT P
100 DATA 48, 56, 28, 63, 59, 54, 1, 3
110 DATA 0, 0, 0, 63, 31, 62, 60, 48
120 DATA 0, 0, 0, 62, 31, 62, 0, 0
200 FOR X = 0 TO 28
210 PRINT AT 10, X; "ABC"
220 PAUSE 10
230 NEXT X
250 GO TO 200
```

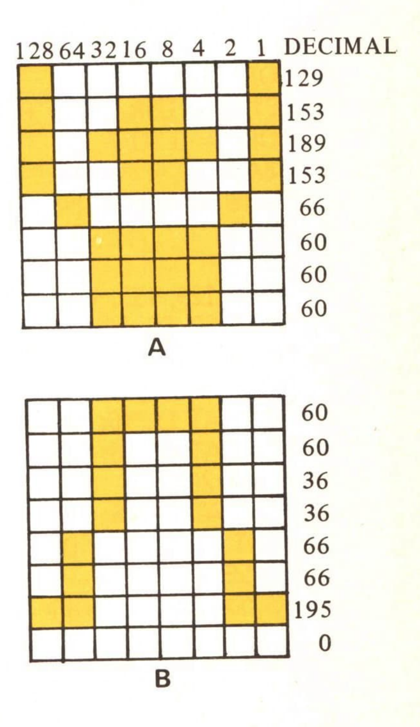
- Si deseamos, también podemos definir un cohete y desplazarlo verticalmente por la pantalla, de abajo hacia arriba.

```
 10 FOR P = 0 TO 1
 20 LET M = USR "A" + P * 8
 30 FOR I = 0 TO 7
 40 READ Q
 50 POKE M + I, Q
 60 NEXT I
 70 NEXT P
100 DATA 12, 30, 45, 63, 63, 30, 12, 0
110 DATA 12, 30, 30, 30, 30, 12, 12, 0
200 FOR Y = 20 TO 2 STEP -1
210 PRINT AT Y, 15; "A"
220 PRINT AT Y + 1, 15; "B"
230 PAUSE 10
240 PRINT AT Y + 1, 15; " "
250 PRINT AT Y + 1, 15; " "
260 NEXT Y
270 GO TO 200
```

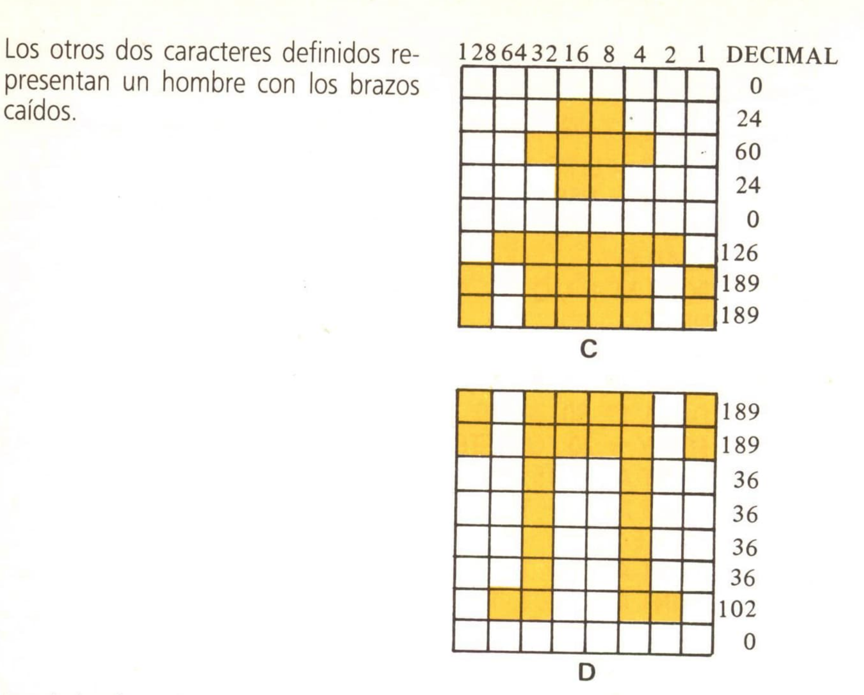
6. El esforzado gimnasta

- Redefiniremos los cuatro caracteres A, B, C y D, y los combinaremos adecuadamente para simular un cierto movimiento.

Los dos primeros caracteres definidos presentan a un hombre con los brazos levantados.



Los otros dos caracteres definidos representan un hombre con los brazos caídos.



Imprimiendo estos cuatro caracteres sucesivamente, conseguiremos la simulación de un movimiento gimnástico.

```
 10 FOR P = 0 TO 3
 20 LET M = USR "A" + P * 8
 30 FOR I = 0 TO 7
 40 READ Q
 50 POKE M + I, Q
 60 NEXT I
 70 NEXT P
100 DATA 129, 153, 189, 153, 66, 60, 60, 60
110 DATA 60, 60, 36, 36, 66, 66, 195, 0
120 DATA 0, 24, 60, 24, 0, 126, 189, 189
130 DATA 189, 189, 36, 36, 36, 36, 102, 0
200 PRINT AT 11, 16; "A"
210 PRINT AT 12, 16; "B"
220 PAUSE 20
230 PRINT AT 11, 16; "C"
240 PRINT AT 12, 16; "D"
250 PAUSE 50
260 GO TO 200
```

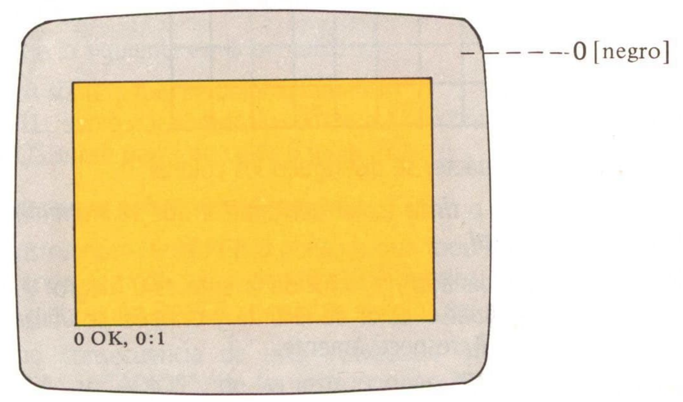
6. Colores y atributos

1. Colores en la pantalla

- Si después de conectar el microordenador tecleamos:

BORDER 0 ENTER

observamos que el borde de la pantalla se pone de color negro.



Esto se debe a que la pantalla está dividida en dos zonas: el ***borde*** y el ***rectángulo de impresión,*** que está formado por 22 filas y 32 columnas, las cuales dan lugar a $22 \times 32 = 704$ posiciones para imprimir caracteres.

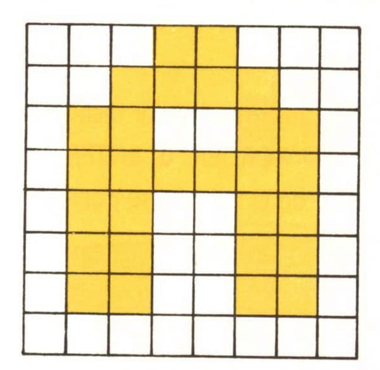
El borde de la pantalla se puede poner en cualquiera de los ocho colores disponibles que posee el microordenador, tecleando los números correspondientes después de BORDER .

0 NEGRO
1 AZUL
2 ROJO
3 MAGENTA
4 VERDE
5 AZUL CLARO
6 AMARILLO
7 BLANCO

Por ejemplo, para poner el borde de color 4 (verde) basta con teclear

BORDER 4 ENTER

- Como se ha visto anteriormente el rectángulo de impresión está dividido en 704 posiciones, y en cada una de ellas se pueden imprimir caracteres. Cada carácter se imprime sobre un cuadrado de 8 × 8 puntos. Por ejemplo, la letra A se imprime de la siguiente forma:



En cada posición de carácter se distinguen los colores:

— el del primer plano o ***tinta*** (color del carácter que se imprime) y
— el del fondo o ***papel.***

Al conectar el microordenador, el color de la tinta es 0 (negro) y el del papel, 7 (blanco). Para variar el color de la tinta y el papel se utilizan las instrucciones INK y PAPER, respectivamente.

Si tecleamos:

INK 2 ENTER
PAPER 6 ENTER

todo lo que se imprima a continuación aparecerá con tinta 2 (rojo) y papel 6 (amarillo).

Si, por ejemplo, tecleamos:

PRINT "BUENOS DIAS" ENTER

aparece esta frase con los colores que habíamos determinado.

Observación: Si a continuación tecleamos ENTER , al comenzar una nueva pantalla todo el rectángulo de impresión se pone del color del papel (en el ejemplo, 6 (amarillo)).

2. Colores en un programa

- Las instrucciones INK y PAPER se pueden utilizar dentro de un programa. Por ejemplo, al ejecutar este programa

```
10  PAPER 6
20  PRINT "HOLA"
30  PRINT PAPER 0; INK 7; "LLLLLLL"
40  PRINT "ADIOS"
50  PAPER 7
```

se obtiene lo siguiente en la pantalla:

— HOLA sobre papel de color 6 (amarillo).
— LLLLLLL sobre papel de color 0 (negro) con tinta de color 7 (blanco).
— ADIOS sobre papel de color 6 (amarillo).

Observaciones sobre este programa

— La instrucción **10 PAPER 6** obliga a que todo lo que se imprima a continuación sea con papel de color 6 (amarillo), hasta que por medio de otra instrucción **PAPER** se cambie el color del papel.

— Como consecuencia de esta instrucción **10 PAPER 6** las palabras "HOLA" y "ADIOS", de las instrucciones **20** y **40,** se imprimen sobre papel 6 (amarillo).

— En la instrucción **30** se ha especificado el color del papel y el de la tinta (papel 0 (negro) y tinta 7 (blanco)), luego la cadena "LLLLLLL" se imprime con estos colores de papel y tinta. Es decir, **PAPER** e **INK** se pueden utilizar en una instrucción **PRINT,** seguidos de punto y coma (;), afectando únicamente a esta instrucción.

— La instrucción **50 PAPER 7,** que pone el color del papel 7 (blanco), que es el normal, sirve para que después de ejecutar el programa el papel vuelva a su color normal. Si no se añade esta instrucción el papel quedaría de color 6 (amarillo), debido a la instrucción **10 PAPER 6.**

PROGRAMAS RESUELTOS

1. Hacer un programa que imprima la palabra "COLORES" en colores de 0 a 7 (el 7 (blanco) no aparecerá por ser el color del papel o color de fondo).

Solución

```
10 FOR I = 0 TO 7
20 PRINT INK I; "COLORES"
30 NEXT I
```

2. Hacer un programa que imprima la frase "EL PAPEL ES DE COLOR" con tinta de color normal (0 (negro)) sobre un papel que varía de 1 a 7.

Solución

```
10 FOR I = 1 TO 7
20 PRINT PAPER I; "EL PAPEL ES DE COLOR"
30 NEXT I
```

3. Hacer un programa que ponga al principio del programa la tinta de color 4 (verde), el papel de color 1 (azul) y el borde de color 2 (rojo). Imprima la frase "ESTOY IMPRIMIENDO EN COLOR". Y, por último, vuelva a los colores normales: tinta, 0 (negro); papel, 7 (blanco) y borde, 7 (blanco).

Solución

```
10 INK 4 : PAPER 1 : BORDER 2
20 PRINT "ESTOY IMPRIMIENDO EN COLOR"
30 INK 0: PAPER 7 : BORDER 7
```

3. Colores en la entrada de datos

- Las instrucciones **INK** y **PAPER** seguidas de punto y coma (;) se pueden poner en una instrucción **INPUT** para que el texto entrecomillado en esta instrucción aparezca en el color que deseemos. Si no se le indica nada el microordenador busca un color que contraste con el color del borde. Al ejecutar este programa

```
10 INPUT INK 1; PAPER 4; "NOMBRE"; A$
20 FOR I = 1 TO 20
30 PRINT A$
40 NEXT I
```

aparecerá en las filas inferiores de la pantalla

[NOMBRE] "[L]"

donde el texto "NOMBRE" de la instrucción **INPUT** aparece con tinta 1 (azul) y papel 4 (verde).

El dato que debemos introducir aparecerá con los colores normales.

4. Colores en alta resolución

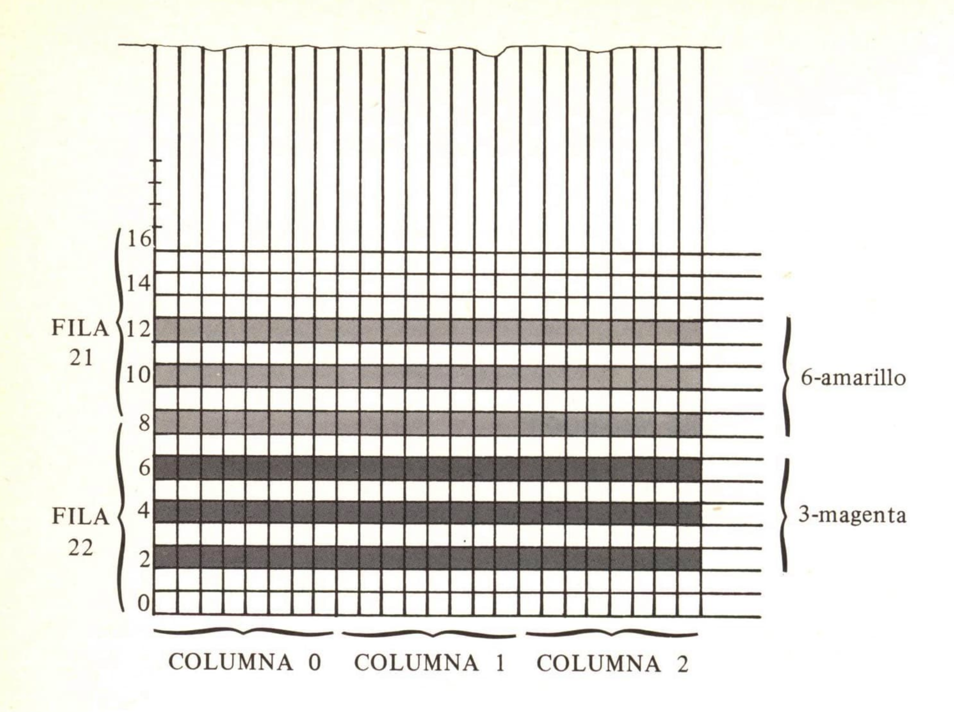
- En las instrucciones **PLOT, DRAW** y **CIRCLE** también se puede especificar el color de la tinta **INK** y del papel **PAPER,** seguidos de punto y coma (;).

Sin embargo, su utilización es problemática, pues al dar color a un pixel se da color (de tinta o papel, según corresponda) a toda la posición de carácter que lo contiene. Luego si en una misma posición de carácter se colorean varios pixels todos éstos tomarán el color del último.

Por ejemplo, si ejecutamos el programa:

```
10  FOR I = 1 TO 6
20  PLOT 0, 2 * I
30  DRAW INK I; 255, 0
40  NEXT I
```

aparecen en la pantalla seis líneas horizontales, que deberían ser cada una de un color distinto, desde el 1 (azul) al 6 (amarillo). Sin embargo, en realidad, las tres inferiores aparecen de color 3 (magenta); esto es debido a que las líneas que se trazan desde I = 1 hasta 3 están situadas sobre las mismas posiciones de carácter, tomando el color de los pixels de la última línea (línea 6). Análogamente sucede con las tres líneas superiores de color 6 (amarillo).



- Si sustituimos la instrucción **10** del programa anterior por:

  10 FOR I = 1 TO 7

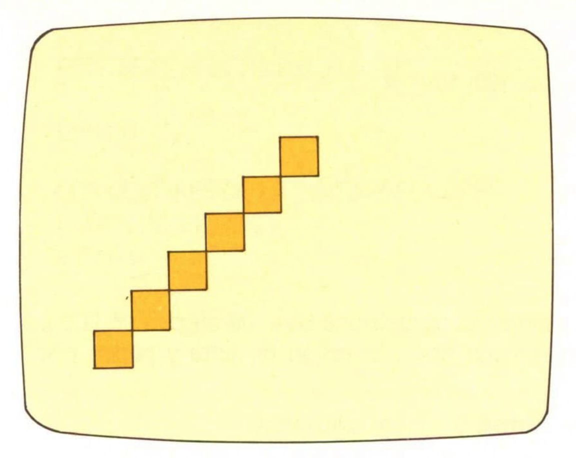
  añadimos como último color el 7 (blanco, color normal de papel). De este modo, las líneas 8, 10, 12 y 14 quedarían en blanco.

- Cuando se utiliza el color de papel **PAPER** también hay que tener cuidado, pues colorea toda la posición de carácter que contiene al pixel.

  Un programa que pone de manifiesto esta característica es:

```
10  PLOT 30, 70
20  DRAW PAPER 4; 40, 40
```

que al ejecutarlo se obtiene la siguiente pantalla:



donde se observa que todas las posiciones de caracteres, correspondientes a los pixels impresos, han tomado el color 4 (verde).

PROGRAMAS RESUELTOS

4. Hacer un programa que imprima una línea recta del 0, 0 al 100, 100 de color 6 (amarillo), y a partir de esta última posición trace otra recta hasta el pixel (150, 100), con tinta 1 (azul) y papel 5 (azul claro).

   Solución

```
10 DRAW INK 6; 100, 100
20 DRAW INK 1; PAPER 5; 50, 0
```

5. Escribir un programa que dibuje 6 círculos, de colores del 1 al 6, con centro en el pixel 100, 100 y cuyos radios aumenten de 12 en 12, siendo el primero de radio 10.

   Solución

```
10 LET A = 0
20 FOR I = 1 TO 6
30 CIRCLE INK I; 100, 100, 10 + A
40 LET A = A + 12
50 NEXT I
```

6. Escribir un programa análogo al anterior pero en lugar de que varíe el color de la tinta varíe el del papel. (Se coloreará toda posición de carácter en que se imprima un pixel.)

Solución

```
10  LET A = 0
20  FOR I = 1 TO 6
30  CIRCLE PAPER I; 100, 100, 10 + A
40  LET A = A + 12
50  NEXT I
```

5. Atributos

- Además de los colores se pueden conseguir una serie de efectos en la pantalla como parpadeo, aumento de brillo, inversión de tinta y papel, y sobreimpresión.

  Estos efectos se llaman ***atributos,*** y son los siguientes:

Atributo	*Significado*	*Función*
FLASH 1	Intermitencia, parpadeo	Va alternando los colores de tinta y de papel.
BRIGHT 1	Brillo	Aumenta la luminosidad de los caracteres.
INVERSE 1	Inverso	Intercambia los colores de tinta y papel en los caracteres.
OVER 1	Sobre	Imprime sobre un carácter sin borrarle (sirve para poner acento a las vocales).

El siguiente programa ilustra estos atributos:

```
10  PRINT FLASH 1; "ESTE ES EL EFECTO DEL FLASH"
20  PRINT
30  PRINT BRIGHT 1; "BRILLO"
40  PRINT
50  PRINT INVERSE 1; "LOS COLORES DE LA TINTA Y EL PAPEL
ESTAN INVERTIDOS"
60  PRINT AT 20, 0; " a e i o u "
70  PRINT AT 20, 0; OVER 1; " ' ' ' ' ' "
```

Al ejecutarlo obtenemos lo siguiente en la pantalla:

```
ESTE ES EL EFECTO DEL FLASH

BRILLO

LOS COLORES DE LA TINTA Y EL PAPEL
ESTAN INVERTIDOS
á é í ó ú
```

— La instrucción **10** imprime la frase ÉSTE ES EL EFECTO DEL FLASH en forma intermitente (parpadeo).

— La instrucción **30** imprime la palabra BRILLO con mayor luminosidad.

— La instrucción **50** imprime la frase LOS COLORES DE LA TINTA Y EL PAPEL ESTÁN INVERTIDOS, poniendo de manifiesto esta propiedad.

— La instrucción **60** imprime las vocales a partir de la fila 20, columna 0 y la instrucción **70** imprime a partir de esa fila y columna los acentos, sin borrar las vocales gracias a la instrucción **OVER 1.**

- Estos atributos también se pueden utilizar en una instrucción **INPUT** (excepto el atributo **OVER**), con los mismos efectos que en una instrucción **PRINT** (aunque sólo afectan a los textos).
- Para desactivar los atributos se sustituye el 1 por el 0:

FLASH 0	**desactiva el parpadeo.**
BRIGHT 0	**desactiva el brillo.**
INVERSE 0	**desactiva el inverso.**
OVER 0	**desactiva la sobreimpresión.**

Los atributos se pueden utilizar fuera de programas. Por ejemplo, si probamos

FLASH 1 ENTER ENTER todo el rectángulo de impresión parpadeará (*)

FLASH 0	ENTER	ENTER	quedará desactivado el parpadeo
BRIGHT 1	ENTER	ENTER	todo el rectángulo de impresión brillará.
BRIGHT 0	ENTER	ENTER	quedará desactivado el brillo.

(*) Si sólo se presiona una vez **ENTER**, el parpadeo y el brillo tendrá efectos parciales sobre la pantalla al principio de su activación.

Con **INVERSE 1** todo lo impreso en pantalla aparecerá con los colores invertidos hasta que se desactive por medio de **INVERSE 0** (si no hay nada escrito en la pantalla no se notarán sus efectos).

Con **OVER 1** no se notan sus efectos hasta que se realice una sobreimpresión. Sus efectos duran hasta que se desactive por medio de **OVER 0.**

Estos atributos se pueden utilizar como instrucciones de un programa, permaneciendo sus efectos aun después de su ejecución, si no se desactivan antes. Los atributos afectan a todas las instrucciones **PRINT** que les sigan, pero no afectan a las instrucciones **INPUT**, pues sus textos entrecomillados aparecen en el borde de la pantalla.

Veamos un programa que tiene en cuenta estas propiedades:

```
10  BRIGHT 1
20  INPUT INVERSE 1; "NOMBRE"; A$
30  PRINT "BRILLANTE"
40  PRINT BRIGHT 0; FLASH 1; "INTERMITENTE", A$
50  PRINT "BRILLANTE"
60  BRIGHT 0
```

— La instrucción **10 BRIGHT 1** activa el brillo y todo lo que se imprima a continuación aparecerá brillante (sólo afectará a **PRINT**).

— La instrucción **20** pide una cadena para A$, apareciendo invertida la palabra "NOMBRE" (el brillo activado en la instrucción **10** no le afecta).

— Las instrucciones **30** y **50** imprimen la palabra BRILLANTE con brillo.

— La instrucción **40** desactiva el brillo y activa el parpadeo (sólo en su ejecución; no afecta al resto del programa), en consecuencia, aparecen las cadenas INTERMITENTE y la que almacena A$, parpadeantes.

7. Hacer un programa que imprima veinte veces la frase "EFECTOS ESPECIALES" y tal que si se introduce

   1 se active el FLASH
   2 se active el BRIGHT
   3 se active el INVERSE

   y si se introduce otro número se imprima "EFECTOS ESPECIALES" sin más.
   Al final de la ejecución deben quedar desactivados.

Solución

```
10  INPUT "UN NUMERO DEL 1 AL 3"; X
20  IF X = 1 THEN FLASH 1
30  IF X = 2 THEN BRIGHT 1
40  IF X = 3 THEN INVERSE 1
50  FOR I = 1 TO 20
60  PRINT "EFECTOS ESPECIALES"
70  NEXT I
80  FLASH 0: BRIGHT 0: INVERSE 0
```

8. Hacer un programa que recuadre en espacios parpadeantes el rectángulo de impresión. Rectificarlo para que el recuadro quede invertido.

Solución

```
10  FLASH 1
20  PRINT "
               "
30  FOR I = 1 TO 20
40  PRINT "   "; AT I, 31; "   "
50  NEXT I
60  PRINT "
               "
70  FLASH 0
```

Para que el recuadro salga invertido basta con sustituir las instrucciones 10 y 70 por:

```
10  INVERSE 1
70  INVERSE 0
```

9. Elaborar un programa que escriba diez veces la palabra "ÑOÑO" utilizando la "N" y el símbolo "~", que se encuentra en la tecla [A].

Solución

```
10 FOR I = 1 TO 10
20 PRINT "NONO"
30 PRINT AT I – 1, 0; OVER 1; "~   ~   "
40 NEXT I
```

6. Atributos en alta resolución

- En las instrucciones **PLOT, DRAW** y **CIRCLE** se pueden incluir los atributos **FLASH** y **BRIGHT,** seguidos de punto y coma (;), y teniendo en cuenta que afectan a toda la posición de carácter que contiene al pixel.

  Es importante tener presente que si **FLASH** o **BRIGHT** se activan en instrucciones anteriores a **PLOT, DRAW** y **CIRCLE,** no afectan a lo que éstas impriman.

Veamos el siguiente programa como ejemplo:

```
10 FLASH 1
20 DRAW 100, 50
30 PLOT BRIGHT 1; 90, 90
40 DRAW BRIGHT 1; 50, 50
50 CIRCLE FLASH 1; 200, 50, 30
60 FLASH 0
```

Al ejecutar el programa se observa:

— Las instrucciones **10** y **60** se pueden suprimir, ya que no producen ningún efecto, pues **10 FLASH 1** no afecta a las instrucciones **PLOT, DRAW** y **CIRCLE.**

— La instrucción **20** traza una línea recta del pixel 0, 0 al 100, 50 sin que le afecte el **FLASH** de la instrucción **10.**

— Las instrucciones **30** y **40** trazan una línea recta desde el pixel 90, 90 al 140, 140 dando brillo a las posiciones de caracteres que contienen a los pixels.

— La instrucción **50** dibuja una circunferencia con centro en el pixel 200, 50 de radio 30, y como le acompaña el atributo **FLASH 1** las posiciones de caracteres que contienen a los pixels que forman la circunferencia, parpadean.

- No ocurre lo mismo con los atributos **INVERSE** y **OVER**, pues en el interior de una instrucción **PLOT, DRAW** o **CIRCLE** sólo afecta al pixel correspondiente y no al resto de la posición de carácter que lo contiene. Los efectos son los siguientes:

INVERSE 1;	pone el pixel de color papel (es como si borrara o no imprimiera).
OVER 1;	cambia el color del pixel (de color papel a color tinta o viceversa) según el color que tuviera el pixel.
INVERSE 1; OVER 1;	no hace nada, solamente cambia la posición donde se imprimió el último pixel.

El programa siguiente muestra estas propiedades:

```
10 PRINT AT 18, 3; "PPPPP"
20 DRAW OVER 1; 255, 175
30 CIRCLE 150, 150, 20
40 STOP
50 CIRCLE INVERSE 1; 150, 150, 20
60 PLOT INVERSE 1; OVER 1; 0, 0
70 DRAW OVER 1; 255, 175
```

— La instrucción **10** imprime la letra p cinco veces (ppppp) a partir de la fila 18, columna 3.

— La instrucción **20** traza una diagonal del pixel 0, 0 al 255, 175 y como lleva el atributo **OVER 1,** al pasar por algún texto impreso, en este caso la tercera "P", cambia los pixeles de color de tinta a color de papel.

— La instrucción **30** imprime una circunferencia con centro en el pixel 150, 150 y de radio 20.

— La instrucción **40** para la ejecución. Si pulsamos **CONT** y **ENTER** continúa la ejecución.

— La instrucción **50,** al contener **INVERSE 1,** borra la circunferencia.

— La instrucción **60** posiciona el último punto impreso en el pixel 0, 0 ***sin marcarlo,*** debido a **INVERSE 1; OVER 1;**.

— La instrucción **70** cambia el color de toda la diagonal, volviéndolo a su color original, con lo que la tercera P vuelve a estar completa.

PROGRAMAS RESUELTOS

10. Hacer un programa que dibuje circunferencias de centro 100, 100 y radios 5, 10, 15, 20, 25, 30 y después las borre una a una.

Solución

```
10  FOR R = 5 TO 30 STEP 5
20  CIRCLE 100, 100, R
30  NEXT R
40  FOR R = 5 TO 30 STEP 5
50  CIRCLE INVERSE 1; 100, 100, R
60  NEXT R
```

11. Hacer un programa que recuadre el rectángulo de impresión con espacios parpadeantes por medio de instrucciones **DRAW.** Hacer otro programa con espacios brillantes.

Solución

Con parpadeo:

```
10  DRAW FLASH 1; 255, 0
20  DRAW FLASH 1; 0, 175
30  DRAW FLASH 1; –255, 0
40  DRAW FLASH 1; 0, –175
```

Con brillo:

```
10  DRAW BRIGHT 1; 255, 0
20  DRAW BRIGHT 1; 0, 175
30  DRAW BRIGHT 1; –255, 0
40  DRAW BRIGHT 1; 0, –175
```

7. Funciones POINT y ATTR

Estas dos funciones investigan las características de un pixel y de una posición carácter, respectivamente. La función **POINT (X, Y)** investiga si el color del pixel de coordenadas X, Y es de tinta o de papel, dando como resultado 1 ó 0, respectivamente. Es decir,

si **POINT (X, Y) = 1,** el pixel X, Y tiene color de tinta;

si **POINT (X, Y) = 0,** el pixel X, Y tiene color de papel.

La función **ATTR (F, C)** investiga, en la posición de carácter de la fila F y columna C, los siguientes atributos:

— color de la tinta,

— color del papel,

— brillo,

— parpadeo.

Esta función da un número entre 0 y 255 de tal forma que:

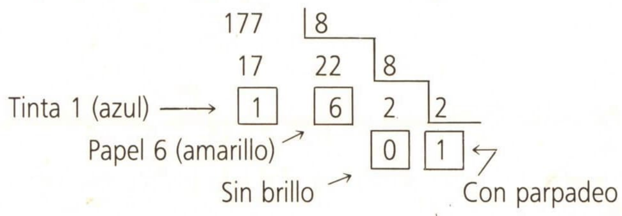
— el resto de su división por 8 da el número del color de la tinta;

— el resto de la división del cociente anterior entre 8 da el número del color del papel;

— el resto de la división del cociente anterior entre 2 indica que hay brillo, si el resto vale 1, y que no lo hay si vale 0;

— el cociente de la división anterior indica que hay parpadeo, si dicho cociente vale 1, y que no lo hay si vale 0.

Por ejemplo, **ATTR (f, C) = 177** da las siguientes informaciones:



8. Resumen

¿Afectan a estas instrucciones los colores y atributos si se activan entes de su ejecución?

	PRINT	INPUT	PLOT	DRAW	CIRCLE
BORDER	SI	SI	SI	SI	SI
INK	SI	NO	SI	SI	SI
PAPER	SI	NO	NO	NO	NO
FLASH	SI	NO	NO	NO	NO
BRIGHT	SI	NO	NO	NO	NO
INVERSE	SI	NO	SI	SI	SI
OVER	SI	NO	SI	SI	SI

¿Afectan a estas instrucciones los colores y atributos si los contienen?

	PRINT	INPUT	PLOT	DRAW	CIRCLE
BORDER	NO	NO	NO	NO	NO
INK	SI	SI	SI	SI	SI
PAPER	SI	SI	SI	SI	SI
FLASH	SI	SI	SI	SI	SI
BRIGHT	SI	SI	SI	SI	SI
INVERSE	SI	SI	SI	SI	SI
OVER	SI	NO	SI	SI	SI

PROGRAMAS RESUELTOS

12. Hacer un programa que vaya poniendo el borde de los distintos colores, haciendo pausas de 200.

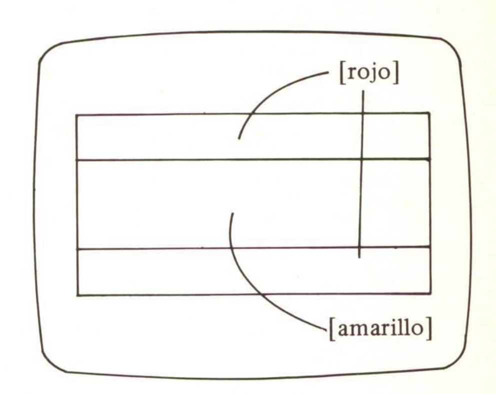
Solución

```
10 FOR I = 0 TO 7
20 BORDER I
30 PAUSE 200
40 NEXT I
```

13. Hacer un programa que dibuje la bandera de España.

Solución

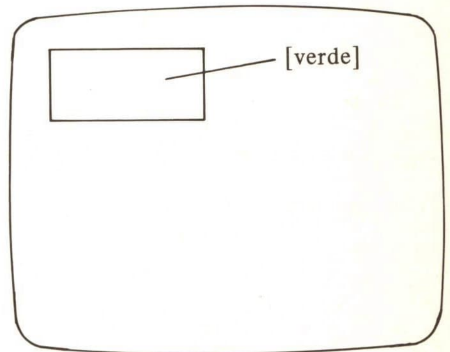
```
 10  PAPER 2
 20  FOR I = 1 TO 32 * 5
 30  PRINT "   ";
 40  NEXT I
 50  PAPER 6
 60  FOR I = 1 TO 32 * 10
 70  PRINT "   ";
 80  NEXT I
 90  PAPER 2
100  FOR I = 1 TO 32 * 5
110  PRINT "   ";
120  NEXT I
130  PAPER 7
```



14. Hacer un programa que dibuje un rectángulo de color verde (papel verde) que ocupe las 5 primeras filas y las 8 primeras columnas.

Solución

```
10  FOR F = 0 TO 4
20  FOR C = 0 TO 7
30  PRINT AT F, C; PAPER 4;"   "
40  NEXT C
50  NEXT F
```



15. Hacer un programa que dibuje un rectángulo del color que queramos, a partir de la posición que se indique y cuya base y altura también podamos determinar (se debe decidir si se dibujan más rectángulos, borrando o no la pantalla).

Solución

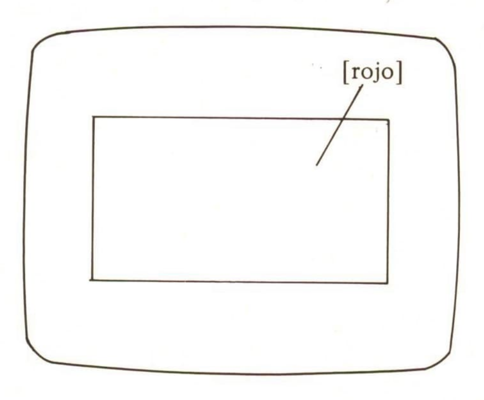
```
 10 INPUT "COLOR (0 – 7)"; CL
 20 INPUT "POSICION INICIAL (FILA Y COLUMNA)"; F1, C1
 30 IF F1 > 21 OR C1 > 31 THEN GO TO 20
 40 INPUT "ALTURA Y BASE"; H, B
 50 IF F1 + H > 21 OR C1 + B > 31 THEN GO TO 40
 60 FOR F = F1 TO F1 + H
 70 FOR C = C1 TO C1 + B
 80 PRINT AT F, C; PAPER CL; "  "
 90 NEXT C
100 NEXT F
110 INPUT "QUIERES DIBUJAR OTRO (S/N)"; A$
120 IF A$ = "N" OR A$ = "n" THEN STOP
130 INPUT "QUIERES BORRAR LA PANTALLA (S/N)"; B$
140 IF B$ = "S" OR B$ = "s" THEN CLS
150 GO TO 10
```

Las instrucciones **30** y **50** hacen que el dibujo no se salga de la pantalla.

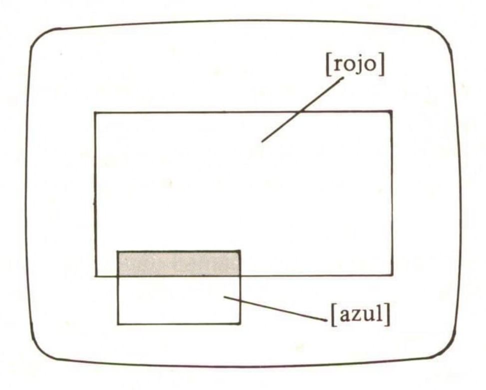
Con las instrucciones **110-150** se decide si se dibujan otros rectángulos y si se borra la pantalla.

Ejecución

RUN ENTER
COLOR (0, 7) 2 ENTER
POSICION INICIAL (FILA Y COLUMNA
) 5 ENTER 6 ENTER
ALTURA Y BASE 10 ENTER 20 ENTER



```
QUIERES DIBUJAR OTRO (s/n) "s"  ENTER
QUIERES BORRAR LA PANTALLA (S/N) "N"  ENTER
COLOR (0, 7) 1  ENTER
POSICION INICIAL FILA Y COLUMNA
) 12  ENTER  15  ENTER
ALTURA Y BASE 8  ENTER  8  ENTER
```




```
QUIERES DIBUJAR OTRO (s/n) "N"  ENTER
9 STOP STATEMENT, 120 : 2
```

Con lo que termina la ejecución.

16. Hacer un programa tal que la variable I vaya tomando alternativamente los valores 1 y 0. Si I = 1 el programa debe dibujar las diagonales de la pantalla y si I = 0 debe borrarlas. El programa no debe tener fin y comenzará con I = 1.

Solución

```
10  LET I = 1
20  PLOT 0, 0
30  DRAW 255, 175
40  PLOT 0, 175
50  DRAW 255, -175
60  IF I = 1 THEN INVERSE 1 : LET I = 0 : GO TO 20
70  INVERSE 0 : LET I = 1 : GO TO 20
```

17. Escribir un programa que dibuje la gráfica de la función seno en color rojo.

Solución

```
10  FOR I = 0 TO 255
20  PLOT INK 2; I, 88 + 41 *SIN (I/128 *PI)
30  NEXT I
```

Las escalas en los ejes de abscisas y ordenadas son las mismas, manteniendo la relación

$$2\pi \rightarrow 256$$

de donde

$$1 \rightarrow 41$$

Los ejes están centrados en el pixel 0, 88.

18. Escribir un programa que dibuje la función seno en color amarillo y la función coseno, en color verde. (Observar lo que sucede en las inmediaciones de los puntos de corte.)

Solución

```
10  FOR I = 0 TO 255
20  PLOT INK 6; I, 88 + 41 *SIN (I/128 *PI)
30  NEXT I
40  FOR I = 0 TO 255
50  PLOT INK 4; I, 88 + 41 *COS (I/128 *PI)
60  NEXT I
```

19. Hacer un programa que dibuje círculos de centro, radio y color variable.

Solución

```
10  INPUT "CENTRO"; X, Y
20  INPUT "RADIO"; R
30  INPUT "COLOR"; C
40  INK C
50  FOR I = 0 TO R
60  CIRCLE X, Y, I
70  NEXT I
80  INPUT "OTRO CIRCULO S/N"; A$
90  IF A$ = "s" OR A$ = "S" THEN GO TO 10
100 INK 0
```

7. Música

1. Algo de música

- La música consiste fundamentalmente en la producción de ciertas frecuencias, durante intervalos de tiempo determinados.

  En el mundo occidental, en la composición de piezas musicales se utiliza la escala de música tradicional, llamada ***escala cromática***, que se compone de doce notas.

  En el teclado de un piano se distinguen dos tipos de teclas: blancas y negras.
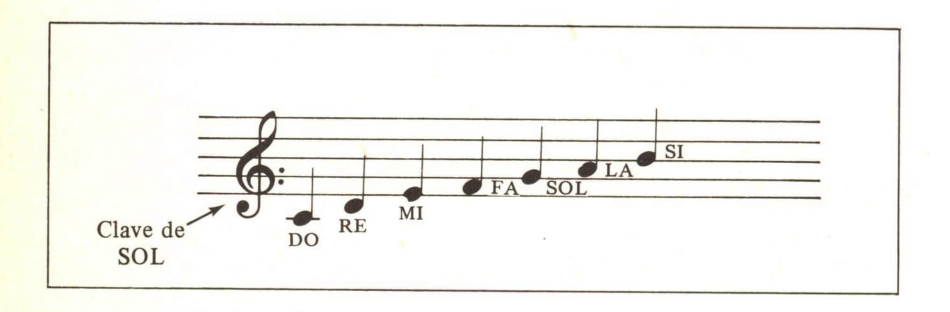


El teclado contiene varios grupos de teclas blancas intercaladas con negras. Las teclas blancas de cada uno de estos grupos corresponden a las notas DO, RE, MI, FA, SOL, LA, SI (en nomenclatura internacional C, D, E, F, G, A, D), que constituyen lo que se llama una ***octava;*** entre dichas teclas blancas se encuentran teclas negras, que corresponden a tonos intermedios (semitonos). Cada semitono se llama ***sostenido*** (#), si toma el nombre de la nota anterior, o bien bemol (β), si adopta el nombre de la nota siguiente. Así, el DO sostenido coincide con el RE bemol.

2. Representación de la música en el pentagrama

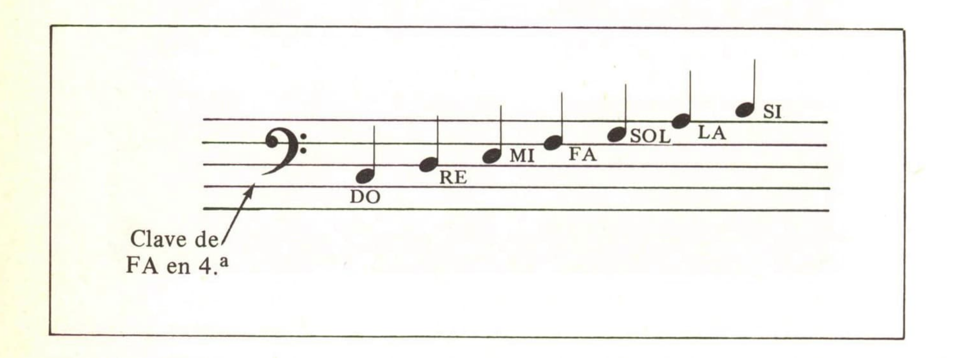
- Las distintas notas se suelen representar en un diagrama llamado ***pentagrama.*** Éste contiene una ***clave,*** que indica la línea en la cual está situada una determinada nota.

  Las claves más frecuentes son ***la clave de sol*** y ***la clave de fa.*** En un pentagrama con clave de SOL, la nota ***sol*** está situada en la segunda línea.



Pentagrama con clave de SOL

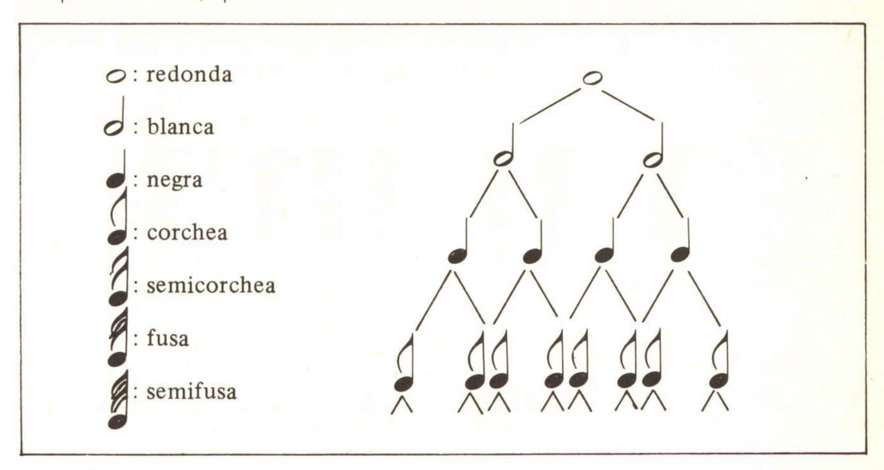
En un pentagrama con clave de FA, la nota FA está situada en la cuarta línea (clave de FA en 4.ª).



Pentagrama con clave de FA

Una pieza musical está caracterizada no solamente por el conjunto de notas que la componen, sino también por la ***duración*** de cada una de ellas, y ésta es la que imprime a la pieza musical un ritmo determinado.

La duración de un sonido se indica en el pentagrama utilizando distintos tipos de notas, que indicamos a continuación:



La duración más larga corresponde a la ***redonda,*** que es la que se toma como referencia. Así, la duración de

— una ***blanca*** es la mitad que la de una redonda; luego una redonda equivale a dos blancas;

— una ***negra*** es la mitad de una blanca; luego una blanca equivale a dos negras;

— etcétera.

Según indica el esquema, 1 nota redonda equivale a 2 blancas, a 4 negras, a 8 corcheas, a 16 semicorcheas, a 32 fusas y a 64 semifusas.

Además de los sonidos y su duración, en una pieza musical intervienen también pausas o silencios; éstos contribuyen también a crear el ritmo de la composición musical.

3. La música en los microordenadores

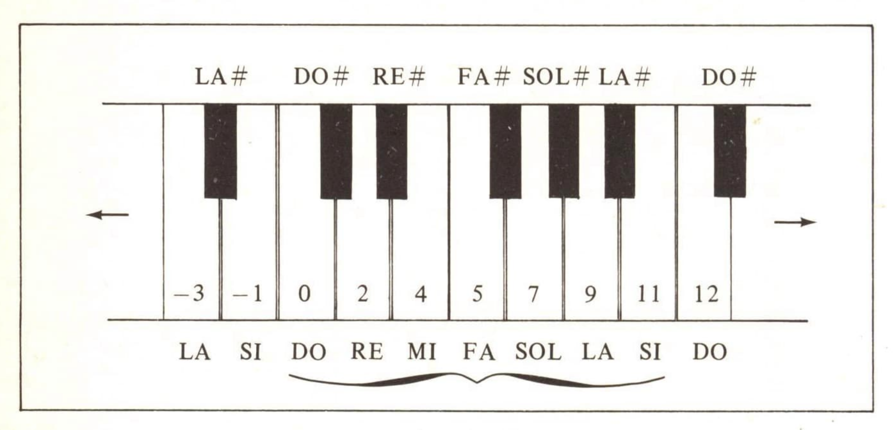
- Los microordenadores con posibilidades musicales pueden controlar como mínimo la frecuencia de los sonidos (altura de las notas) y la duración de las mismas. Esto se consigue en el microordenador ZX Spectrum por medio de la instrucción

BEEP d, n

siendo d y n valores numéricos que corresponden a la duración y a la altura de la nota, respectivamente.

La duración ***d*** de una nota puede variar entre 0,00125 y 10 segundos.

La altura de una nota se indica con un número comprendido entre –60 y 69. Los números del 0 al 11, corresponden a las alturas de las notas de la escala cromática.



Escala cromática

- Por el siguiente programa aplicamos la instrucción **BEEP** para generar la escala cromática veinte veces.

```
10  FOR N = 1 TO 28
20  FOR P = 0 TO 11
30  BEEP 0.25, P
40  NEXT P
50  PAUSE 25
60  NEXT N
```

Cada nota emitida adquiere una duración de 0.25 segundos.

La instrucción **50 PAUSE 25** introduce una pausa de medio segundo entre dos escalas consecutivas (*).

- Una octava sin semitonos puede generarse mediante este otro programa en el que la instrucción 70 reestablece el archivo situado en la instrucción

(*) Para conseguir un mayor volumen de sonido conectar la salida EAR del ZX Spectrum con la entrada MIC de un casete y presionar PLAY. Esta conexión se puede hacer con el cable que sirve para grabar y reproducir programas.

100; esta instrucción contiene los números correspondientes a las notas DO, RE, MI, FA, SOL, LA, SI, DO.

```
 10  FOR N = 1 TO 12
 20  FOR I = 1 TO 8
 30  READ P
 40  BEEP 0.25, P
 50  NEXT I
 60  PAUSE 25
 70  RESTORE
 80  NEXT N
100  DATA 0, 2, 4, 5, 7, 9, 11, 12
```

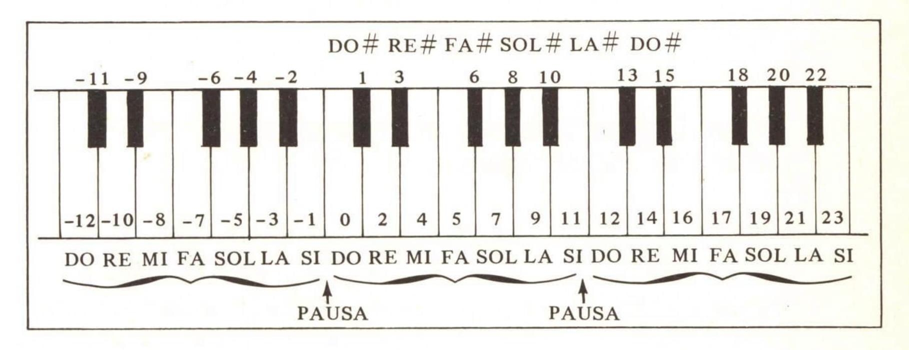
También aquí cada sonido producido por la instrucción 40 dura 0.25 segundos.

Este programa repite la escala doce veces **(10 FOR N = 1 TO 12).**

PROGRAMAS RESUELTOS

1. Hacer un programa que genere las tres escalas cromáticas representadas en el teclado de la figura. Introducir una pausa de 2 segundos entre cada escala.

```
10  REM TRES ESCALAS CROMATICAS
20  FOR N = –12 TO 23
30  BEEP 0.30, N
40  IF N = –1 OR N = 11 THEN PAUSE 100
50  NEXT N
```

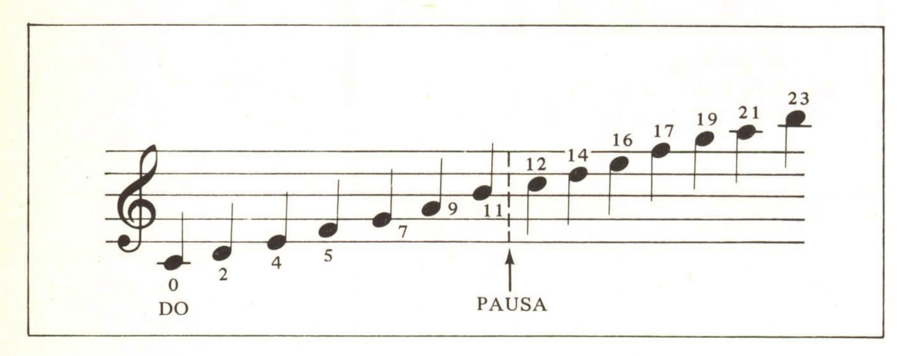


Solución

— A cada nota se le ha dado una duración de 0.30 segundos (instrucción **30**).

— La instrucción **40** introduce la pausa de 2 segundos, aproximadamente, entre dos escalas sucesivas.

2. En el pentagrama, cada nota va acompañada del número que hay que poner en la instrucción **BEEP** para que sea generada. Elaborar un programa que reproduzca dichas notas, introduciendo entre las dos escalas una pausa de 2 segundos.



Solución

```
10  REM DOS ESCALAS
20  FOR I = 0 TO 14
30  READ N
40  BEEP 0.40, N
50  IF N = 11 THEN PAUSE 100
60  NEXT I
70  DATA 0, 2, 4, 5, 7, 9, 11, 12, 14, 16, 17, 19, 21, 23
```

— En este programa, a cada nota se le ha dado una duración de 0.40 segundos (instrucción **40**).

— Los valores de las notas de las dos escalas se encuentran en la instrucción **70 DATA**.

4. Cómo convertir el teclado del microordenador en un «teclado de órgano»

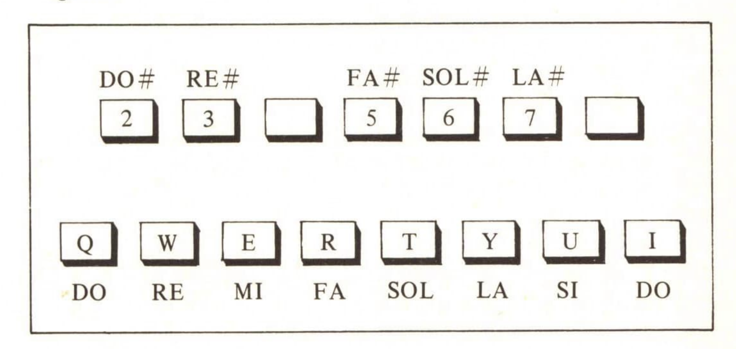
- El programa que transcribimos a continuación transforma parte del teclado del microordenador en parte del teclado de un órgano.

  Una vez cargado este programa en la memoria, si se ejecuta (pulsar **RUN** y **ENTER**) se puede interpretar la melodía de algunas canciones con las teclas de las dos filas superiores del teclado.

```
 10  LET A$ = INKEY$
 20  LET P = 0.75
 30  IF A$ = "Q" THEN BEEP P, 0
 40  IF A$ = "2" THEN BEEP P, 1
 50  IF A$ = "W" THEN BEEP P, 2
 60  IF A$ = "3" THEN BEEP P, 3
 70  IF A$ = "E" THEN BEEP P, 4
 80  IF A$ = "R" THEN BEEP P, 5
 90  IF A$ = "5" THEN BEEP P, 6
100  IF A$ = "T" THEN BEEP P, 7
110  IF A$ = "6" THEN BEEP P, 8
120  IF A$ = "Y" THEN BEEP P, 9
130  IF A$ = "7" THEN BEEP P, 10
140  IF A$ = "U" THEN BEEP P, 11
150  IF A$ = "I" THEN BEEP P, 12
160  GO TO 10
```

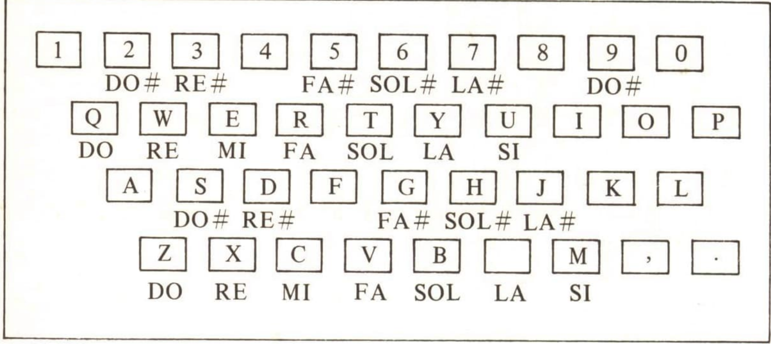
— La instrucción **10** lee el teclado, y el carácter de la tecla pulsada se almacena en A$, de tal manera que si en el momento de la ejecución se pulsa, por ejemplo la letra Q (instrucción **30 IF A$ = "Q" THEN BEEP P, 0**), se genera la nota correspondiente al número 0, o sea, la nota DO, con una duración de 0.75 segundos.

— Las instrucciones **30** a **150** hacen que al pulsar las teclas Q, 2, W, 3, E, R, 5, T, 6, Y, 7, U, I, se genere la escala cromática de un órgano.



Se puede ampliar este teclado añadiendo instrucciones al programa, a efectos de conseguir notas más graves (con números menores que 0), y notas más agudas (con números mayores que 12). Para ello, conviene tener en cuenta que entre las notas MI y FA no existe semitono, lo mismo que entre las notas SI y DO.

3. Convertir el teclado del microordenador en teclado de órgano de tal manera que permita tocar dos escalas acromáticas completas. Asignar a las dos filas inferiores la escala más baja.



Solución

```
  5 REM TECLADO DE ORGANO CON DOS ESCALAS
ACROMATICAS SEGUIDAS
 10 LET A$ = INKEY$
 20 LET P = 0.75
 30 IF A$ = "Z" THEN BEEP P, 0
 40 IF A$ = "S" THEN BEEP P, 1
 50 IF A$ = "X" THEN BEEP P, 2
 60 IF A$ = "D" THEN BEEP P, 3
 70 IF A$ = "C" THEN BEEP P, 4
 80 IF A$ = "Y" THEN BEEP P, 5
 90 IF A$ = "G" THEN BEEP P, 6
100 IF A$ = "B" THEN BEEP P, 7
110 IF A$ = "H" THEN BEEP P, 8
120 IF A$ = "N" THEN BEEP P, 9
130 IF A$ = "J" THEN BEEP P, 10
140 IF A$ = "M" THEN BEEP P, 11
150 IF A$ = "Q" THEN BEEP P, 12
160 IF A$ = "2" THEN BEEP P, 13
170 IF A$ = "W" THEN BEEP P, 14
180 IF A$ = "3" THEN BEEP P, 15
190 IF A$ = "E" THEN BEEP P, 16
200 IF A$ = "R" THEN BEEP P, 17
210 IF A$ = "5" THEN BEEP P, 18
220 IF A$ = "T" THEN BEEP P, 19
230 IF A$ = "6" THEN BEEP P, 20
240 IF A$ = "Y" THEN BEEP P, 21
250 IF A$ = "7" THEN BEEP P, 22
260 IF A$ = "U" THEN BEEP P, 23
270 GO TO 10
```

— La escala más grave se consigue interpretar con las teclas de las dos filas inferiores (instrucciones **30** a **140**).

— La escala más aguda se consigue interpretar con las teclas de las dos filas superiores (instrucciones **150** a **260**).

5. Un programa que reproduce una melodía

- Si se ejecuta el siguiente programa (pulsar **RUN** y **ENTER**) se obtiene una melodía que recuerda en algo la canción popular mexicana ***Cielito lindo.***

```
5  REM CIELITO LINDO
10 INPUT "INTRODUCE EL NUMERO DE NOTAS"; N
20 FOR I = 1 TO N
30 READ L
40 BEEP 0.40, L
50 NEXT I
70 DATA 12, 12, 9, 11, 7, 12, 9, 11, 7, 12, 12, 9, 11, 7, 5, 2, 2,
11, 11, 11
80 DATA 11, 9, 7, 5, 2, 2, 4, 5, 7, 7, 7, 7, 7, 5, 4, 2, 0, 16, 14, 12,
9, 9, 14, 14, 12
90 DATA 16, 12, 12, 7, 9, 7, 9, 9, 7, 17, 17, 14, 14, 11, 7, 11, 11,
9, 9, 7, 5, 4, 2, 0, 0
```

Antes de ejecutar este programa hay que contar el número de notas que componen la melodía; este número coincide con el número de datos incluidos en todas las instrucciones **DATA** (**70,** en este ejmplo).

Para escuchar la melodía hay que ejecutar el programa (pulsar **RUN** y **ENTER**) y, a continuación, introducir el valor 70 para N.

6. Melodías con ritmo

- Si se ha ejecutado el programa anterior se ha podido comprobar que la melodía carecía de todo ritmo. Esto es debido a que todas las notas suenan durante el mismo tiempo (0.40 segundos).

  Para conseguir que el microordenador interprete una melodía con ritmo, es necesario asignar a cada nota la duración que le corresponda en la melodía. Esto se consigue haciendo una pequeña modificación en el programa

anterior, introduciendo una nueva variabe D, que controla la duración de cada nota.

```
  5  REM CIELITO LINDO CON RITMO
 10  INPUT "INTRODUCE EL NUMERO DE NOTAS"; N
 20  FOR I = 1 TO N
 30  READ D, L
 40  BEEP D, L
 50  NEXT I
 60  DATA 0.4, 12, 0.4, 12, 0.4, 9, 0.8, 11, 0.4, 7, 0.4, 12, 0.4, 12,
0.4, 9, 0.8, 11, 0.4, 7
 70  DATA 0.4, 12, 0.4, 12, 0.4, 9, 0.8, 11, 0.4, 7, 0.4, 5, 1.6, 2,
0.4, 11, 0.4, 11, 0.4, 11
 80  DATA 0.8, 11, 0.4, 9, 0.4, 7, 0.4, 5, 0.8, 2, 0.4, 4, 0.4, 5, 0.4,
7, 0.4, 7, 0.4, 7, 0.8, 7, 0.4, 5, 0.4, 4
 90  DATA 0.4, 2, 1.6, 0, 1.2, 16, 0.8, 14, 0.4, 12, 2, 9
100  DATA 1.2, 14, 0.8, 14, 0.4, 12, 0.4, 16, 1.6, 12, 0.4, 7
110  DATA 0.8, 9, 0.4, 7, 0.4, 9, 0.4, 9, 0.4, 7, 0.4, 17, 0.4, 17, 0.8,
14, 0.4, 11, 0.4, 7
120  DATA 0.4, 11, 0.4, 11, 0.8, 9, 0.4, 7, 0.4, 5, 0.2, 4, 0.2, 2, 1.2,
0
```

En las instrucciones **DATA** a cada número indicativo de cada nota precede otro número (0.4, 0.8, etc.) que da la duración de la misma en décimas de segundo.

Si se ejecuta este programa (pulsar **RUN** y **ENTER**) se comprueba que interpreta la melodía de ***Cielito lindo*** con el ritmo que le corresponde, mejorando considerablemente la interpretación anterior.

Este mismo programa puede reproducir otras melodías, modificando las instrucciones **DATA**, tal como se puede comprobar en los siguientes ejemplos.

- ***Noche de paz, de Franz Gruber***

```
DATA 0.6, 7, 0.2, 9, 0.4, 7, 1.2, 4, 0.6, 7, 0.2, 9, 0.4, 7, 1.2, 4, 0.8, 14, 0.4,
14, 1.2, 11
DATA 0.8, 12, 0.4, 12, 1.2, 7, 0.8, 9, 0.4, 9, 0.6, 12, 0.2, 11, 0.4, 9, 0.6,
7, 0.2, 9, 0.4, 7, 1.2, 4
DATA 0.8, 9, 0.4, 9, 0.6, 12, 0.2, 11, 0.4, 9, 0.6, 7, 0.2, 9, 0.4, 7, 1.2, 4,
0.8, 14, 0.4, 16, 0.6, 17, 0.2, 14, 0.4, 11
DATA 1.2, 12, 1.2, 16, 0.6, 12, 0.2, 7, 0.4, 4, 0.6, 7, 0.2, 5, 0.4, 2, 1.6,
0
```

- ***Canción de cuna, de J. Brahms***

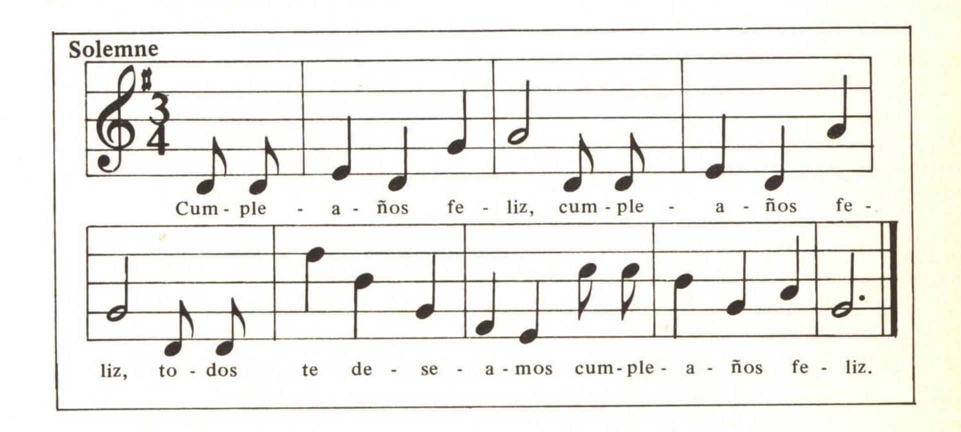
DATA 0.2, 4, 0.2, 4, 0.8, 7, 0.2, 4, 0.2, 4, 0.8, 7, 0.2, 4, 0.2, 7, 0.4, 12, 0.6, 11, 0.2, 9
DATA 0.4, 9, 0.4, 7, 0.2, 2, 0.2, 4, 0.4, 5, 0.4, 2, 0.2, 2, 0.2, 4, 0.8, 5, 0.2, 2, 0.2, 5, 0.2, 11, 0.2, 9, 0.4, 7, 0.4, 11
DATA 0.8, 12, 0.2, 0, 0.2, 0, 0.8, 12, 0.2, 9, 0.2, 5, 0.8, 7, 0.2, 4, 0.2, 0, 0.4, 5, 0.4, 7, 0.4, 9
DATA 0.4, 4, 0.4, 7, 0.2, 0, 0.2, 0, 0.8, 12, 0.2, 9, 0.2, 5, 0.8, 7, 0.2, 4, 0.2, 0, 0.2, 5, 0.1, 7, 0.1, 5, 0.4, 4, 0.4, 2, 0.8, 0

- ***A coger el trébole (popular)***

DATA 0.3, 11, 0.1, 12, 0.2, 14, 0.2, 16, 0.3, 16, 0.1, 14, 0.2, 11, 0.2, 14, 0.3, 14, 0.1, 12, 0.8, 9, 0.2, 12
DATA 0.2, 16, 0.2, 16, 0.4, 14, 0.3, 11, 0.1, 12, 0.2, 14, 0.2, 16, 0.3, 16, 0.1, 14, 0.2, 11, 0.8, 14, 0.3, 14, 0.1, 12, 0.2, 9, 0.2, 12
DATA 1.2, 11, 0.3, 1, 0.1, 12, 0.2, 16, 0.3, 16, 0.1, 14, 0.2, 11, 0.2, 14, 0.3, 14, 0.1, 12, 0.2, 9, 0.2, 12, 0.2, 16, 0.2, 16, 0.4, 14
DATA 0.3, 11, 0.1, 12, 0.2, 14, 0.2, 16, 0.3, 16, 0.1, 14, 0.2, 11, 0.2, 14, 0.3, 14, 0.1, 12, 0.4, 9, 0.4, 12, 0.4, 11

7. Cómo pasar la música de un pentagrama al microordenador

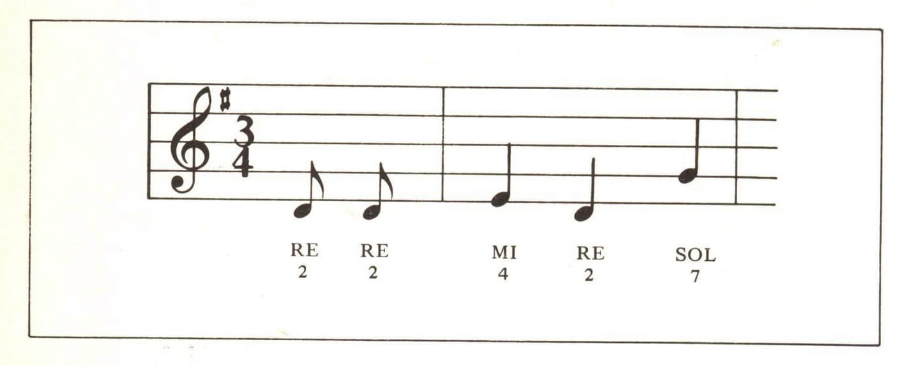
- Para pasar la música escrita en un pentagrama a la memoria del microordenador no hay que saber teoría musical. Solamente es necesario tener en cuenta unas pocas reglas, que expondremos a continuación, eligiendo como ejemplo la popular melodía de ***Cumpleaños feliz.***



- ***Reglas***

1. ***Se identifica cada nota teniendo en cuenta la clave que figura al principio del pentagrama.***

   En este ejemplo, como clave es de SOL, esta nota se encuentra en la 2.ª línea. Luego la melodía empieza con la nota RE.



2. ***Se asigna a cada nota su número correspondiente.***

   El símbolo # ***(sostenido)*** que aparece en la 5.ª línea (nota FA) al principio del pentagrama, indica que todas las notas FA que se encuentran en el pentagrama son sostenidos; les corresponderán entonces el número 6, en lugar del 5.

3. ***Se asigna a cada nota un segundo número que especifica su duración.***

   Este segundo número es el que conferirá a la pieza musical su ritmo característico.

   Como esta melodía es ***solemne,*** asignaremos a la ***nota negra*** (♩) un tiempo bastante largo, por ejemplo, seis décimas de segundo (0.6); en consecuencia, la duración de la ***nota corchea*** (♪) será de 0.3 segundos, y la de la ***blanca*** (𝅗𝅥), 1.2 segundos.

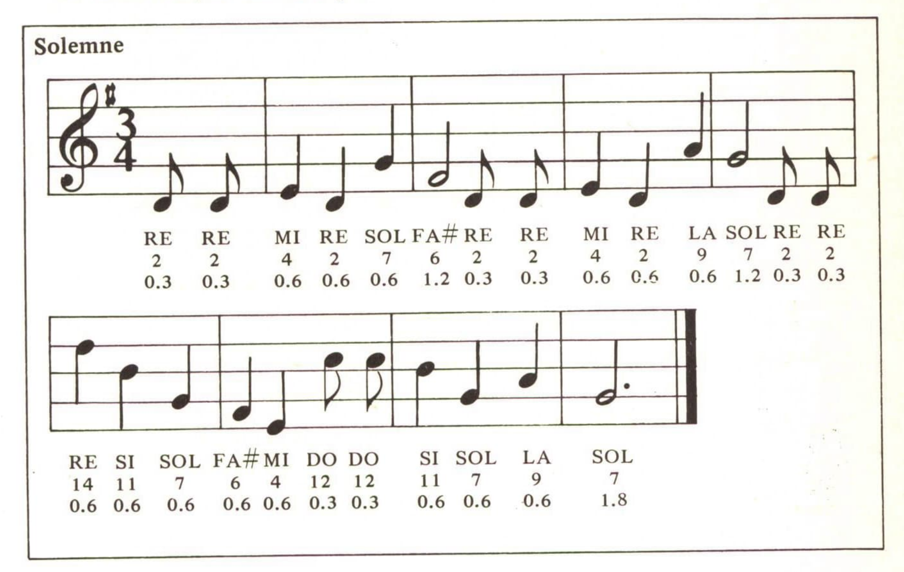


   Un punto a la derecha de una nota aumenta su duración en la mitad de su valor. Así, la duración de la nota 𝅗𝅥. es 1.2 + 0.6 = 1.8 segundos.

4. Cumplidas las reglas anteriores, se teclea el programa visto en el apartado anterior, transcribiendo en instrucciones DATA los dos valores asignados a cada nota (reglas 3 y 2).

- ***Apiicación de las reglas***

A continuación transcribimos otra vez el pentagrama, indicando el nombre de cada nota, su número y su duración.
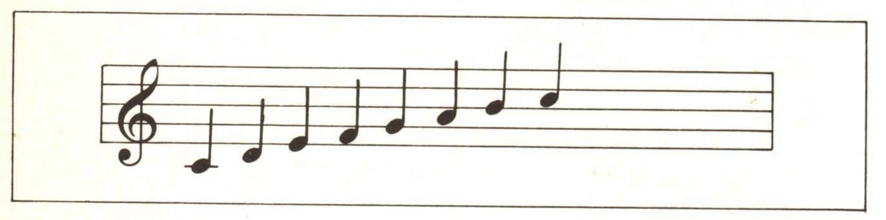


Con estos datos, podemos ya escribir el programa que interpreta la melodía de ***Cumpleaños feliz.***

```
5  REM CUMPLEAÑOS FELIZ
10 INPUT "INTRODUCE EL NUMERO DE NOTAS"; N
20 FOR I = 1 TO N
30 READ D, L
40 BEEP D, L
50 NEXT I
60 DATA 0.3, 2. 0.3, 2. 0.6, 4, 0.6, 2, 0.6, 7, 1.2, 6, 0.3, 2, 0.3, 2,
   0.6, 4, 0.6, 2, 0.6, 9
70 DATA 1.2, 7, 0.3, 2, 0.3, 2, 0.6, 14, 0.6, 11, 0.6, 7, 0.6, 6, 0.6,
   4, 0.3, 12, 0.3, 12, 0.6, 11, 0.6, 7, 0.6, 9, 1.8, 7
```

Para conseguir la interpretación de la melodía, teclear **RUN** y **ENTER**, e introducir el valor 25 para N (número de notas).

4. Hacer un programa que interprete varias veces las notas del siguiente pentagrama.



Solución

— Las notas escritas en el pentagrama son:

DO	RE	MI	FA	SOL	LA	SI	DO
0	2	4	5	7	9	11	12

— A estas notas les corresponden los números escritos debajo de sus nombres.
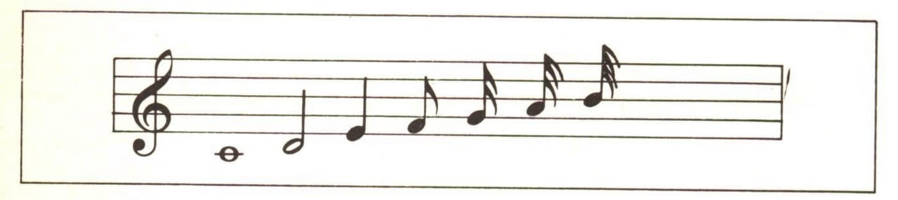
— Como todas las notas son negras (♩), tienen la misma duración.

— El programa que interpreta estas notas es el siguiente:

```
  5 REM INTERPRETA LAS NOTAS VARIAS VECES
 10 INPUT "INTRODUCE NUMERO DE VECES"; V
 20 FOR I = 1 TO V
 30 FOR K = 1 TO 8
 40 READ L
 50 BEEP 0.25, L
 60 NEXT K
 70 PAUSE 100
 80 RESTORE
 90 NEXT I
100 DATA 0, 2, 4, 5, 7, 9, 11, 12
```

Entre cada interpretación se interpone una pausa de unos dos segundos (instrucción **70 PAUSE 100**).

5. Elaborar un programa que interprete varias veces la escala escrita en el pentagrama, interponiendo entre cada interpretación una pausa de unos tres segundos.



Solución

```
  5 REM INTERPRETA LA ESCALA EN FORMA "ACELERADA"
 10 INPUT "INTRODUCE NUMERO DE VECES" V
 20 FOR I = 1 TO V
 30 LET D = 4
 40 FOR K = 1 TO 7
 50 READ L
 60 BEEP D, L
 70 LET D = D - D/2
 80 NEXT K
 90 PAUSE 150
100 RESTORE
110 NEXT I
120 DATA 0, 2, 4, 5, 7, 9, 11
```

En esta escala las notas que siguen a la primera tienen una duración igual a la mitad de la anterior. Es decir, si a la primera se le da una duración de 4 segundos, la duración de las siguientes es la indicada a continuación:



Para conseguir esta duración para cada nota se ha incluido la instrucción **40 LET D = 4** y la **80 LET D = D − D/2** que reduce a la mitad la duración de la nota siguiente.

6. Hacer un programa que interprete varias veces la escala

DO	RE	MI	FA	SOL	LA	SI	DO
0	2	4	5	7	9	11	12
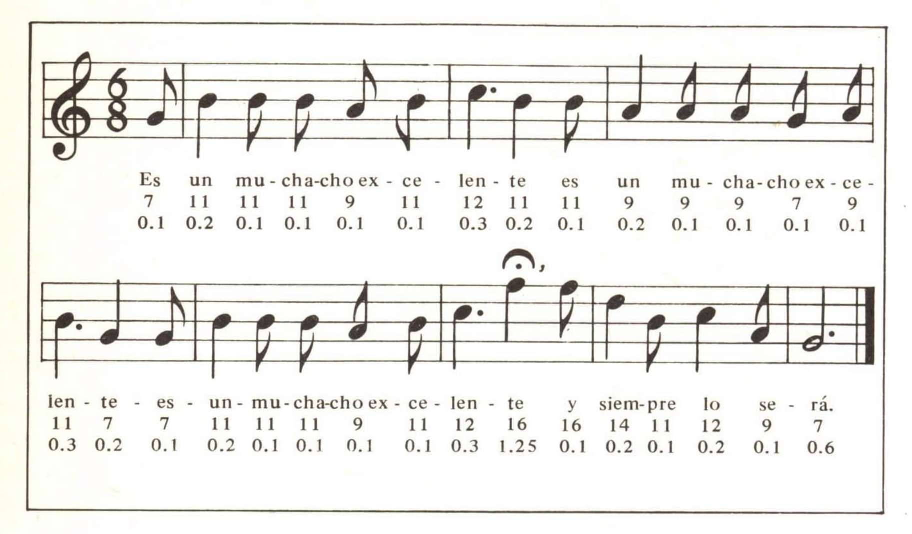
dando a cada nota una duración igual a la de la anterior menos la quinta parte de ésta, o sea, D − D/5.

Solución

```
  5 REM INTERPRETA ESCALA DESCENDENTE CON UN
    RITMO DADO POR UNA FORMULA
 10 INPUT "INTRODUCE NUMERO DE VECES"; V
 20 FOR I = 1 TO V
 30 LET D = 2
 40 FOR K = 1 TO 7
 50 READ L
 60 BEEP D, L
 70 LET D = D - D/5
 80 NEXT K
 90 PAUSE 100
100 RESTORE
110 NEXT I
120 DATA 11, 9, 7, 5, 4, 2, 0
```

7. **Es un muchacho excelente**

Hacer un programa que interprete la popular melodía de la canción **Es un muchacho excelente.**



Solución

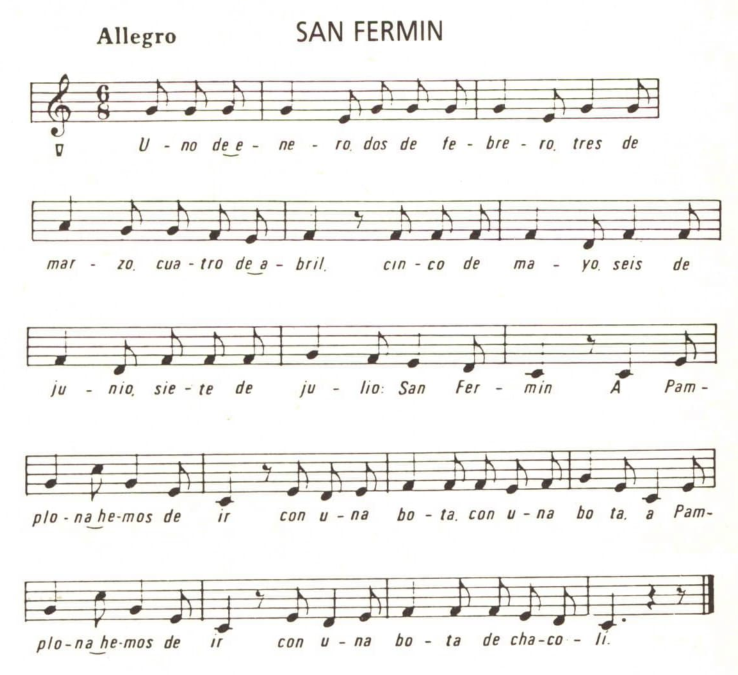
```
 5 REM ES UN MUCHACHO EXCELENTE
10 FOR I = 1 TO 30
20 READ D, L
30 BEEP D, L
40 NEXT I
50 DATA 0.1, 7, 0.2, 11, 0.1, 11, 0.1, 9, 0.1, 11, 0.3, 12, 0.2, 11, 0.1,
   11, 0.2, 9, 0.1, 9, 0.1, 9, 0.1, 7, 0.1, 9
60 DATA 0.3, 11, 0.2, 7, 0.1, 7, 0.2, 11, 0.1, 11, 0.1, 11, 0.1, 9, 0.1,
   11
70 DATA 0.3, 12, 1.25, 16, 0.1, 16, 0.2, 14, 0.1, 11, 0.2, 12, 0.1, 9,
   0.6, 7
```

En esta melodía hay una nota que tiene encima un símbolo especial, (𝄐) llamado ***calderón.*** La duración de la nota afectada por este símbolo se puede prolongar todo el tiempo que se desee. En este ejemplo, la duración que la hemos asignado es 1.25 segundos, valor que supera al del resto de las notas.

8. Algunas canciones populares españolas

Transcribimos a continuación las notas de algunas melodías populares españolas, y las instrucciones DATA necesarias para que el microordenador las interprete con el programa visto anteriormente:

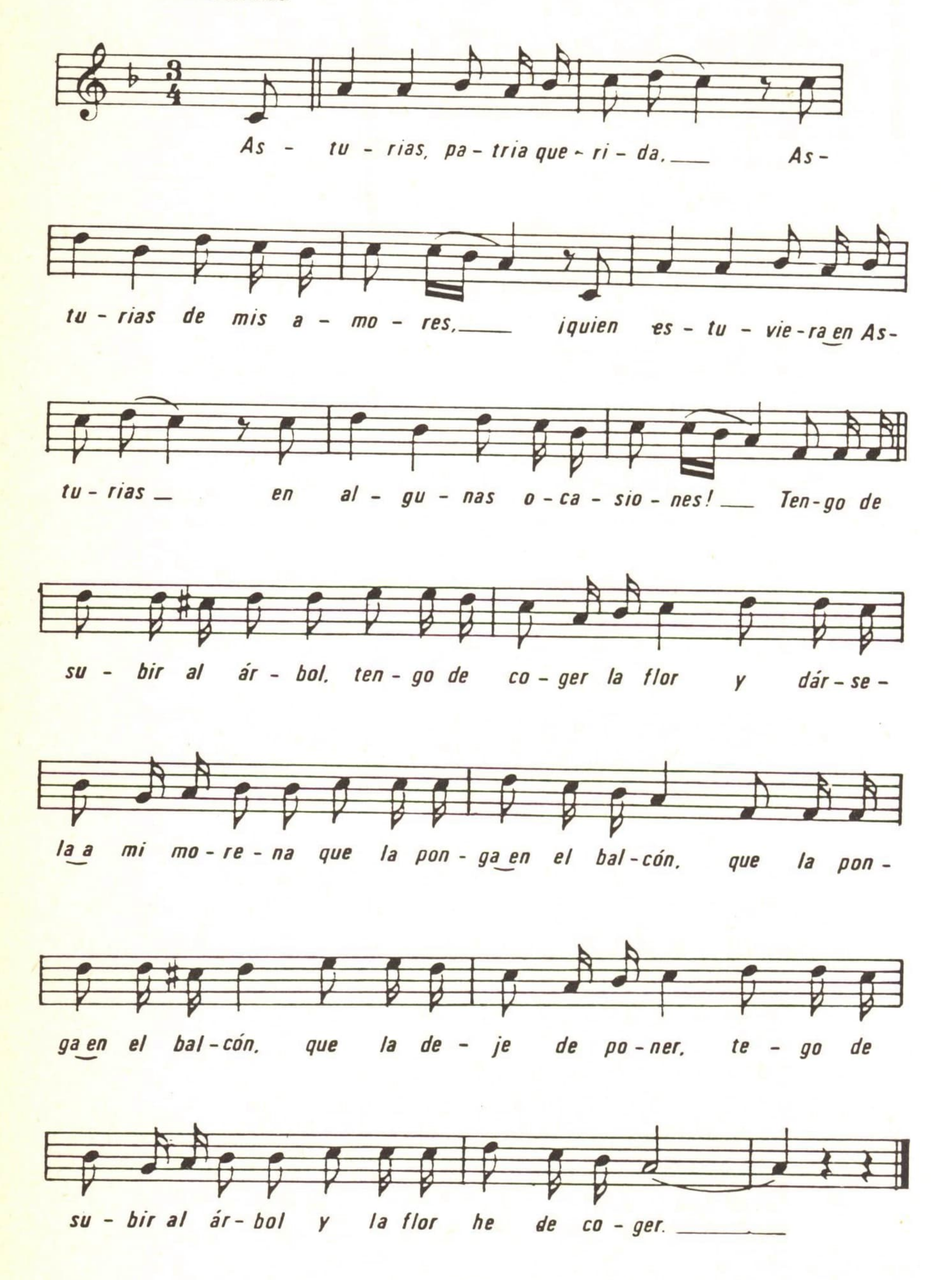
```
 5  REM INTERPRETA MELODIAS
10  INPUT "INTRODUCE EL NUMERO
    DE NOTAS"; N
20  FOR I = 1 TO N
30  READ D, L
40  BEEP D, L
50  NEXT I
60  DATA...
70  DATA...
     ...
```




```
60  DATA 0.15,7, 0.15,7, 0.15,7, 0.3,7, 0.15,4, 0.15,7, 0.15,7, 0.15,7,
    0.3,7, 0.15,4, 0.3,7, 0.15,7
70  DATA 0.3,9, 0.15,7, 0.15,7, 0.15,5, 0.15,4, 0.3,5, 0.15,5, 0.15,5,
    0.15,5, 0.3,5, 0.15,2, 0.3,5, 0.15,5
80  DATA 0.3,5, 0.15,2, 0.15,5, 0.15,5, 0.15,5, 0.3,7, 0.15,5, 0.3,4,
    0.15,2, 0.3,0, 0.3,0, 0.15,4, 0.3,7, 0.15,12, 0.3,7
90  DATA 0.15,4, 0.3,0, 0.15,4, 0.15,2, 0.15,4, 0.3,5, 0.15,5, 0.15,5,
    0.15,4, 0.15,5, 0.3,7, 0.15,4, 0.3,0, 0.15,4
100 DATA 0.3,7, 0.15,12, 0.3,7, 0.15,4, 0.3,0, 0.15,4, 0.15,2, 0.15,4,
    0.3,5, 0.15,5, 0.15,5, 0.15,4, 0.15,2, 0.45,0
```
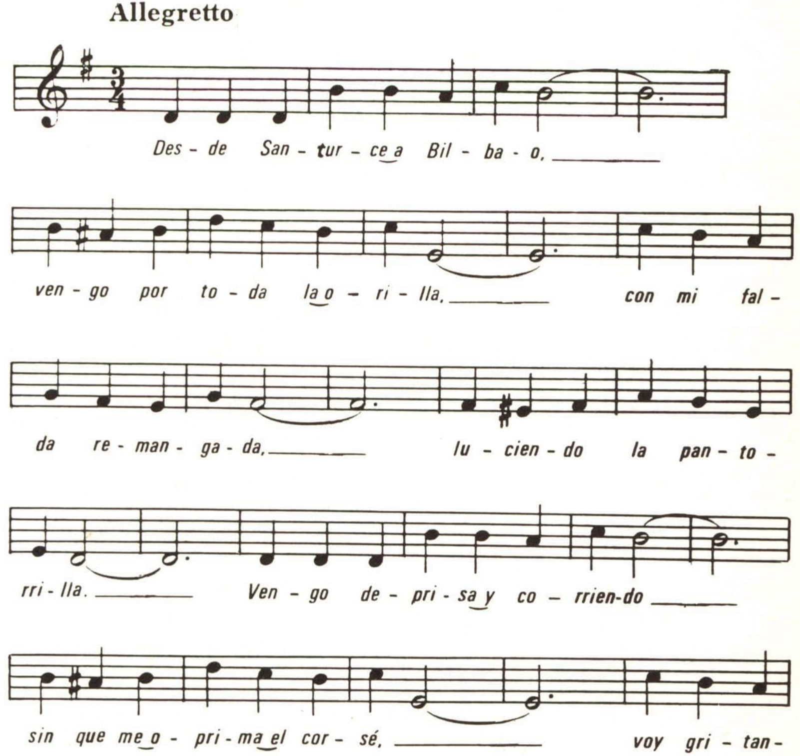
ASTURIAS, PATRIA QUERIDA

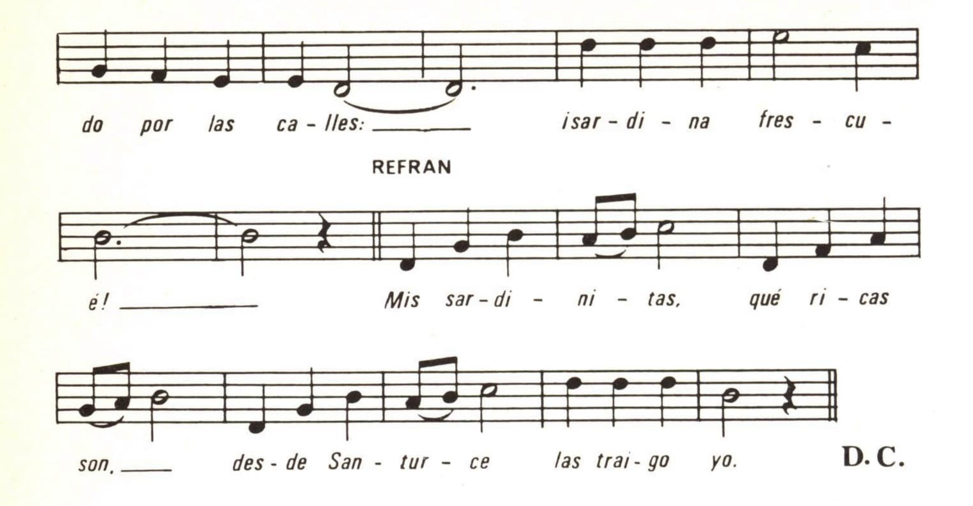
Moderato



```
60  DATA 0.2, 0, 0.4, 9, 0.4, 9, 0.2, 10, 0.1, 9, 0.1, 10, 0.2, 12, 0.2,
    14, 0.6, 12, 0.2, 12
70  DATA 0.4, 14, 0.4, 10, 0.2, 14, 0.1, 12, 0.1, 10, 0.2, 12, 0.1, 12,
    0.1, 10, 0.6, 9, 0.2, 0, 0.4, 9, 0.4, 9, 0.2, 10, 0.1, 9, 0.1, 10
80  DATA 0.2, 12, 0.2, 14, 0.6, 12, 0.2, 12, 0.4, 14, 0.4, 10, 0.2, 14,
    0.1, 12, 0.1, 10, 0.2, 12, 0.1, 12, 0.1, 10, 0.4, 9, 0.2, 5, 0.1, 5, 0.1, 5
90  DATA 0.2, 14, 0.1, 14, 0.1, 13, 0.2, 14, 0.2, 14, 0.2, 16, 0.1, 16,
    0.1, 14, 0.2, 12, 0.1, 9, 0.1, 10, 0.4, 12, 0.2, 14, 0.1, 14, 0.1, 12
100 DATA 0.2, 10, 0.1, 7, 0.1, 9, 0.2, 10, 0.2, 10, 0.2, 12, 0.1, 12,
    0.1, 12, 0.2, 14, 0.1, 12, 0.1, 12, 0.4, 9, 0.2, 5, 0.1, 5, 0.1, 5
110 DATA 0.2, 14, 0.2, 14, 0.1, 13, 0.4, 14, 0.2, 16, 0.1, 16, 0.1, 14,
    0.2, 12, 0.1, 9, 0.1, 10, 0.4, 12, 0.2, 14, 0.1, 14, 0.1, 1
120 DATA 0.2, 10, 0.1, 7, 0.1, 9, 0.2, 10, 0.2, 10, 0.2, 12, 0.1, 12,
    0.1, 12, 0.2, 14, 0.1, 12, 0.1, 10, 1.2, 9
```
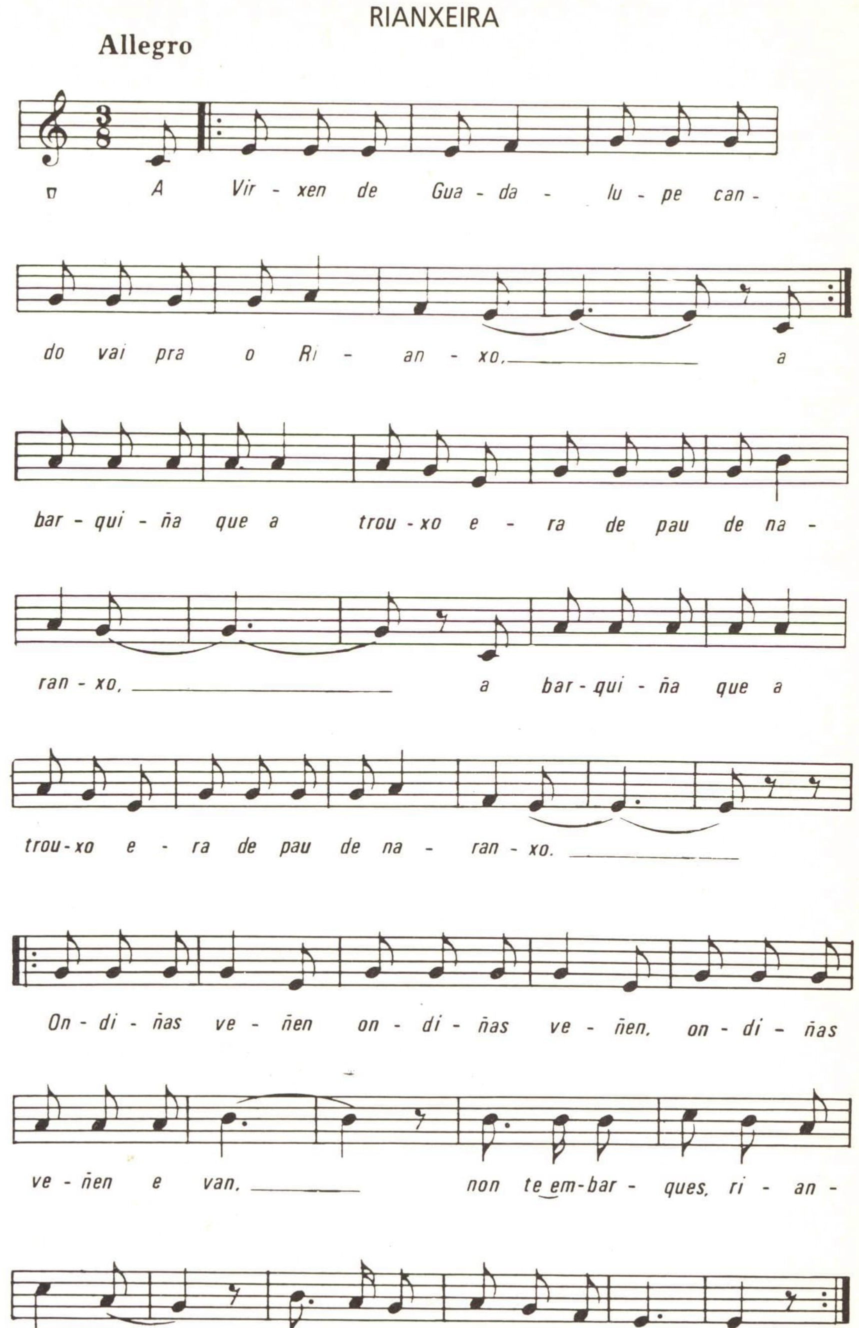






La del segundo me llama,
la del primero, también,
la del tercero me dice:
—¿A cómo las vende "usté"?
Yo le contesto que a cuatro,
ella me dice que a tres,
cojo la cesta y le digo:
¡Sardina frescué!

(Al Refrán)

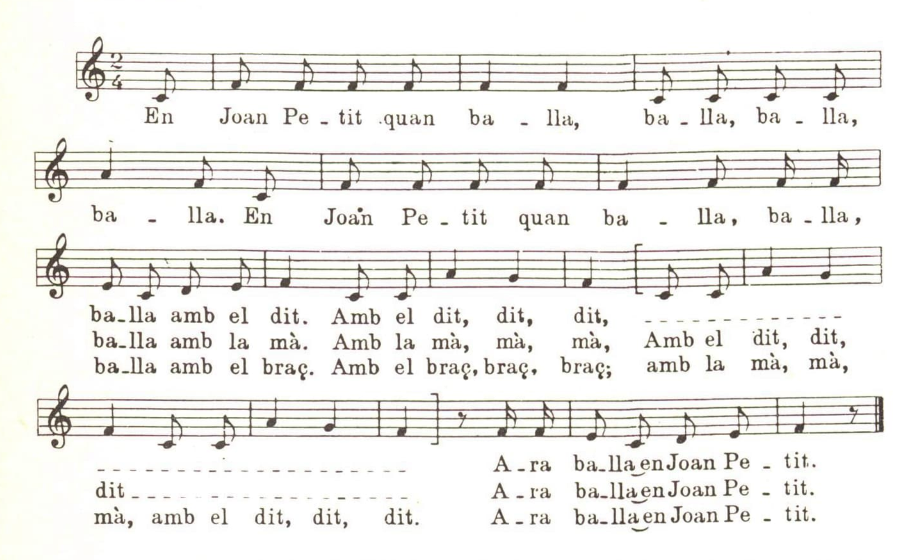
```
60  DATA 0.2, 2, 0.2, 2, 0.2, 2, 0.2, 11, 0.2, 11, 0.2, 9, 0.2, 12, 1,
    11
70  DATA 0.2, 11, 0.2, 10, 0.2, 11, 0.2, 14, 0.2, 12, 0.2, 11, 0.2, 12,
    1, 4, 0.2, 12, 0.2, 11, 0.2, 9
80  DATA 0.2, 7, 0.2, 6, 0.2, 4, 0.2, 7, 1, 6, 0.2, 6, 0.2, 5, 0.2, 6,
    0.2, 9, 0.2, 7, 0.2, 4
90  DATA 0.2, 4, 1, 2, 0.2, 2, 0.2, 2, 0.2, 2, 0.2, 11, 0.2, 11, 0.2, 9,
    0.2, 12, 1, 11
100 DATA 0.2, 11, 0.2, 10, 0.2, 11, 0.2, 14, 0.2, 12, 0.2, 11, 0.2, 12,
    1.2, 4, 0.2, 12, 0.2, 11, 0.2, 9
110 DATA 0.2, 7, 0.2, 6, 0.2, 4, 0.2, 4, 1, 2, 0.2, 14, 0.2, 14, 0.2, 14,
    0.4, 16, 0.2, 12
120 DATA 1.2, 11, 0.2, 2, 0.2, 7, 0.2, 11, 0.1, 9, 0.1, 11, 0.4, 12, 0.2,
    2, 0.2, 6, 0.2, 9
130 DATA 0.1, 7, 0.1, 9, 0.4, 11, 0.2, 2, 0.2, 7, 0.2, 11, 0.1, 9, 0.1,
    11, 0.4, 12, 0.2, 14, 0.2, 14, 0.2, 14, 1, 11
```


RIANXEIRA
Allegro
A Vir - xen de Gua - da - lu - pe can -
do vai pra o Ri - an - xo, a
bar - qui - ña que a trou - xo e - ra de pau de na -
ran - xo, a bar - qui - ña que a
trou - xo e - ra de pau de na - ran - xo.
On - di - ñas ve - ñen on - di - ñas ve - ñen, on - di - ñas
ve - ñen e van, non te em - bar - ques, ri - an -
xei - ra, que te vas a ma - re - ar.

```
60  DATA 0.2,0, 0.2,4, 0.2,4, 0.2,4, 0.2,4, 0.4,5, 0.2,7, 0.2,7, 0.2,7,
    0.2,7, 0.2,7, 0.2,7
70  DATA 0.2,7, 0.4,9, 0.4,5, 1.2,4, 0.2,0, 0.2,9, 0.2,9, 0.2,9, 0.2,9,
    0.4,9
80  DATA 0.2,9, 0.2,7, 0.2,4, 0.2,7, 0.2,7, 0.2,7, 0.2,7, 0.4,11, 0.4,9,
    1.2,7
90  DATA 0.2,0, 0.2,9, 0.2,9, 0.2,9, 0.2,9, 0.4,9, 0.2,9, 0.2,7, 0.2,4,
    0.2,7, 0.2,7, 0.2,7
100 DATA 0.2,7, 0.4,9, 0.4,5, 1.2,4, 0.2,7, 0.2,7, 0.2,7
110 DATA 0.4,7, 0.2,4, 0.2,7, 0.2,7, 0.2,7, 0.4,7, 0.2,4, 0.2,7, 0.2,7,
    0.2,7
120 DATA 0.2,9, 0.2,9, 0.2,9, 1.2,11, 0.2,11, 0.2,11, 0.2,11, 0.2,12,
    0.2,11, 0.2,9
130 DATA 0.4,12, 0.2,9, 0.4,7, 0.2,11, 0.2,9, 0.2,7, 0.2,9, 0.2,7,
    0.2,5, 1.2,4
```

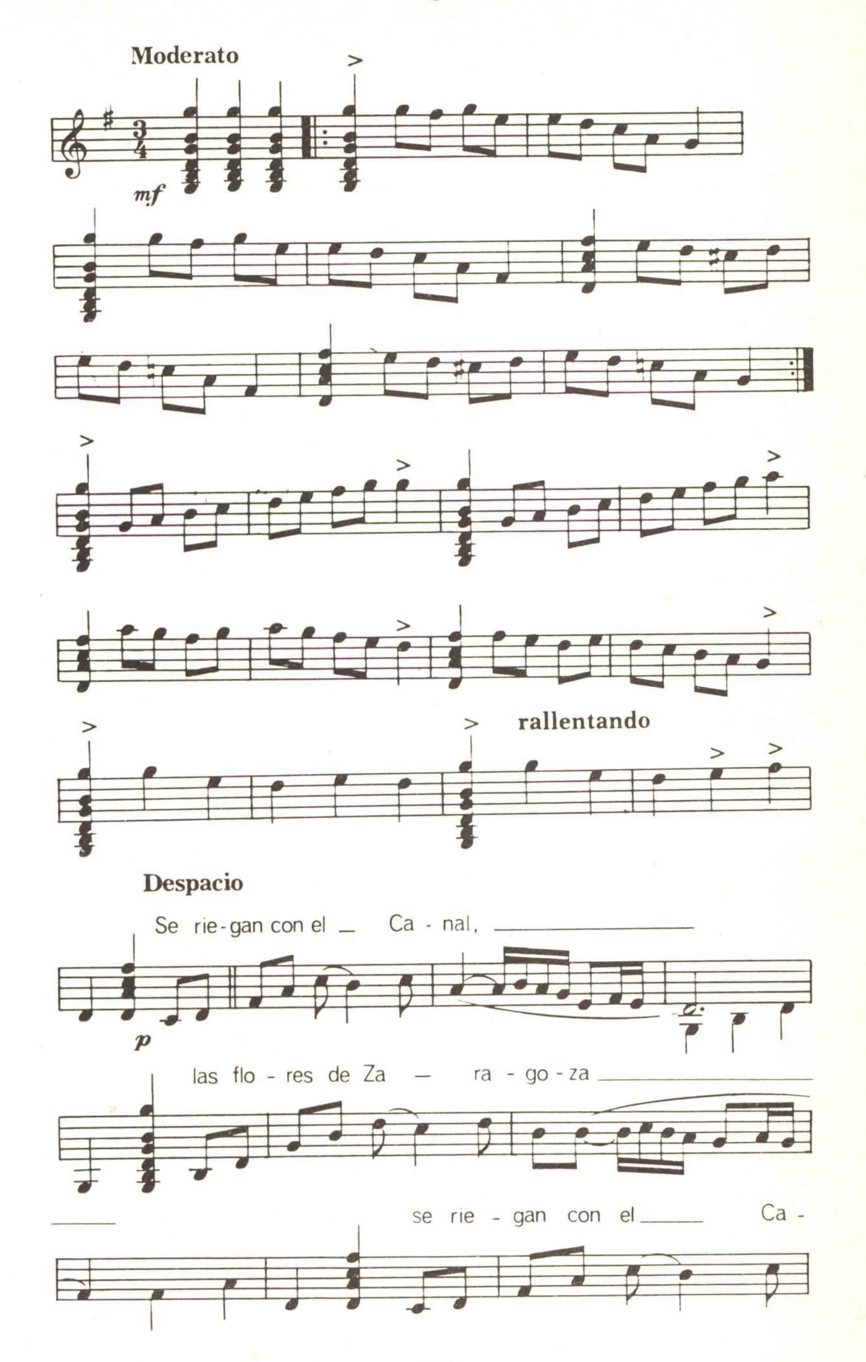
EN JOAN PETIT
Popular catalá

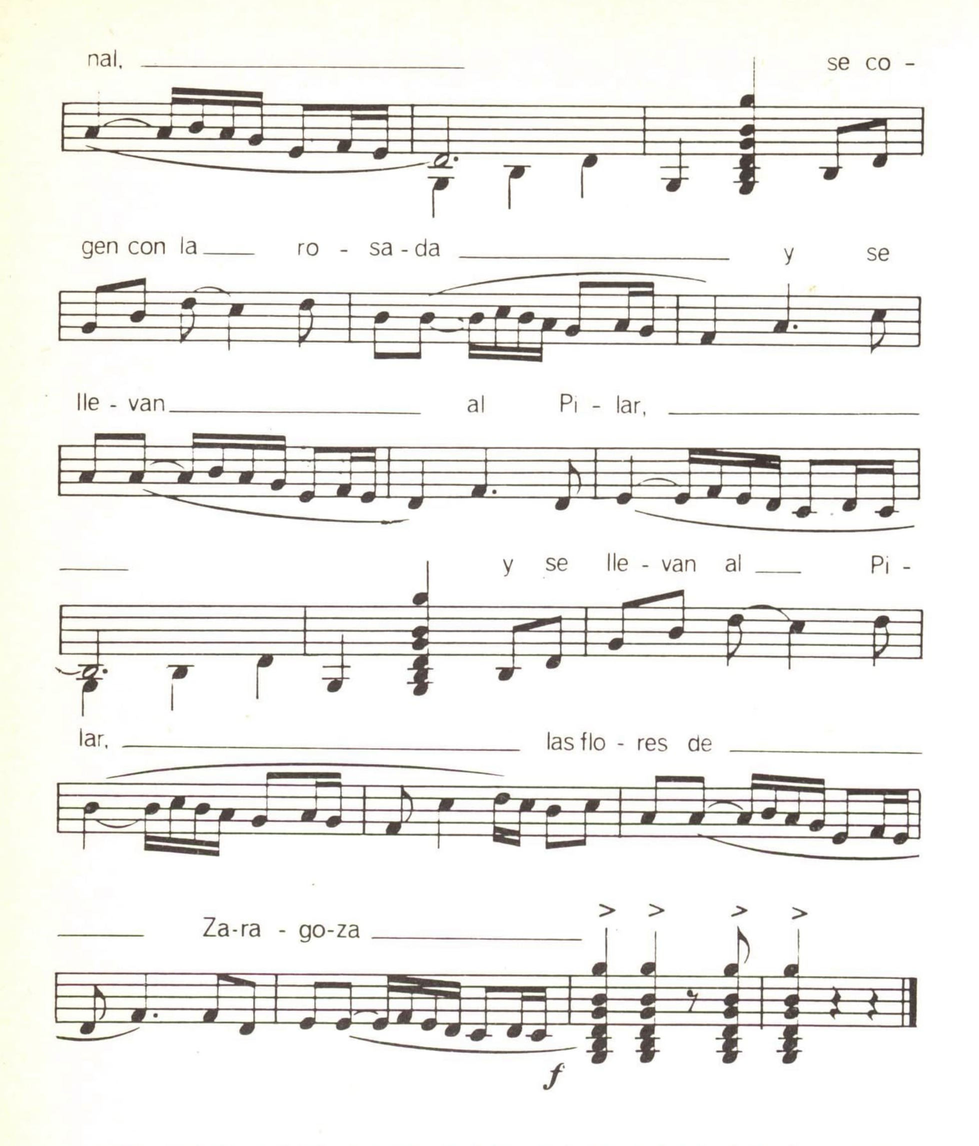
Popular catalana




```
60  DATA 0.2,0, 0.2,5, 0.2,5, 0.2,5, 0.2,5, 0.4,5, 0.4,5, 0.2,0, 0.2,0,
    0.2,0, 0.2,0, 0.4,9
70  DATA 0.2,5, 0.2,0, 0.2,5, 0.2,5, 0.2,5, 0.2,5, 0.4,5, 0.2,5, 0.1,5,
    0.1,5, 0.2,4, 0.2,0
80  DATA 0.2,0, 0.2,4, 0.4,5, 0.2,0, 0.2,0, 0.4,9, 0.4,7, 0.4,5, 0.2,0,
    0.2,0, 0.4,9, 0.4,7
90  DATA 0.4,5, 0.2,0, 0.2,0, 0.4,9, 0.4,7, 0.4,5, 0.1,5, 0.1,5, 0.2,4,
    0.2,0, 0.2,2, 0.2,4, 0.4,5
```

LAS FLORES DE ZARAGOZA
Jota aragonesa






```
60 DATA 0.4,19, 0.4,19, 0.4,19, 0.4,19, 0.2,19, 0.2,18, 0.2,19,
   0.2,16, 0.2,16, 0.2,14, 0.2,12, 0.2,9, 0.4,7
70 DATA 0,4.19, 0.2,19, 0.2,18, 0.2,19, 0.2,16, 0.2,16, 0.2,14,
   0.2,12, 0.2,9, 0.4,6, 0.4,14, 0.2,16, 0.2,14, 0.2,13, 0.2,14
80 DATA 0.2,16, 0.2,14, 0.2,12, 0.2,9, 0.4,6, 0.4,14, 0.2,16, 0.2,14,
   0.2,13, 0.2,14, 0.2,16, 0.2,14, 0.2,12, 0.2,9, 0.4,7
90 DATA 0.4,19, 0.2,7, 0.2,9, 0.2,11, 0.2,12, 0.2,14, 0.2,16, 0.2,18,
   0.2,19, 0.4,19, 0.4,19, 0.2,7, 0.2,9, 0.2,11, 0.2,12, 0.2,14,
   0.2,16, 0.2,18, 0.2,19, 0.4,21
```

```
100 DATA 0.4,18, 0.2,21, 0.2,19, 0.2,18, 0.2,19, 0.2,21, 0.2,19,
    0.2,18, 0.2,16, 0.4,14, 0.4,18, 0.2,18, 0.2,16, 0.2,14, 0.2,16,
    0.2,14, 0.2,12, 0.2,11, 0.2,9, 0.4,7
110 DATA 0.4,19, 0.4,19, 0.4,16, 0.4,14, 0.4,16, 0.4,14, 0.4,19,
    0.4,19, 0.4,16, 0.4,14, 0.4,16, 0.4,18
120 DATA 0.4,2, 0.4,2, 0.2,0, 0.2,2, 0.2,6, 0.2,7, 0.2,12, 0.4,11,
    0.2,12, 0.3,9, 0.1,11, 0.1,9, 0.1,7, 0.2,4, 0.1,6, 0.1,4, 0.4,2,
    0.4,-1, 0.4,2
130 DATA 0.4,-5, 0.4,-5, 0.4,-1, 0.2,2, 0.2,7, 0.2,11, 0.2,14, 0.4,12,
    0.2,14, 0.2,11, 0.3,11, 0.1,12, 0.1,11, 0.1,9, 0.2,7, 0.1,9, 0.1,7
140 DATA 0.4,6, 0.4,6, 0.4,9, 0.4,2, 0.4,2, 0.2,0, 0.2,2, 0.2,6, 0.2,2,
    0.2,12, 0.4,11, 0.2,12
150 DATA 0.5,9, 0.1,11, 0.1,9, 0.1,7, 0.2,4, 0.1,6, 0.1,4, 0.4,-5,
    0.4,-1, 0.4,2, 0.4,-5, 0.4,-5, 0.2,-1, 0.2,2
160 DATA 0.2,7, 0.2,11, 0.2,14, 0.4,12, 0.2,14, 0.2,11, 0.3,11,
    0.1,12, 0.1,11, 0.1,9, 0.2,7, 0.1,9, 0.1,7, 0.4,6, 0.6,9, 0.2,12
170 DATA 0.2,9, 0.3,9, 0.1,11, 0.1,9, 0.1,7, 0.2,4, 0.1,6, 0.1,4, 0.4,2,
    0.6,6, 0.2,2, 0.5,4, 0.1,6, 0.1,4, 0.1,2, 0.2,0, 0.1,2, 0.1,0
180 DATA 0.4,-5, 0.4,-1, 0.4,2, 0.4,-5, 0.4,-5, 0.2,-1, 0.2,2, 0.2,7,
    0.2,11, 0.2,14, 0.4,12, 0.2,14
190 DATA 0.5,11, 0.1,12, 0.1,11, 0.1,9, 0.2,7, 0.1,9, 0.1,7, 0.2,6,
    0.4,12, 0.1,14, 0.1,12, 0.2,11, 0.2,12, 0.2,9, 0.3,9, 0.1,11, 0.1,9,
    0.1,7, 0.2,4, 0.1,6, 0.1,4
200 DATA 0.2,2, 0.5,6, 0.2,6, 0.2,2, 0.2,4, 0.3,4, 0.1,6, 0.1,4, 0.1,2,
    0.2,0, 0.1,2, 0.1,0, 0.4,7, 0.4,7, 0.2,7, 0.4,7
```



APÉNDICE I

BASIC DEL ZX Spectrum

Palabras BASIC	Función que realiza
ABS X	Da el valor absoluto del número X.
AT F, C	Sitúa el cursor en la posición correspondiente a la fila F y a la columna C.
ATN X	Calcula el arco cuya tangente es X. (El arco calculado se expresa en radianes.)
CHR$ X	Da el carácter del código ASCII correspondiente al número X. (X es un entero comprendido entre 32 y 128.)
CLS	Borra lo escrito en la pantalla pero no lo almacenado en la memoria del microordenador.
LOAD "XXX"	Carga en la memoria del microordenador un programa grabado en un casete con el nombre XXX.
CONT	Continúa la ejecución de un programa interrumpida por la instrucción **STOP.**
CODE X$	Da el número del código ASCII correspondiente al primer carácter de la cadena X$.
COS X	Calcula el coseno del ángulo X, expresado en radianes.
SAVE "XXX"	Graba en un casete con el nombre XXX un programa almacenado en la memoria del microordenador.
DATA	Almacena los datos que serán leídos por una instrucción **READ.**

Palabras BASIC	Función que realiza
DEF FNA (X)	Define una función numérica de nombre FNA y de variable X. (Para definir otras funciones cambiar la letra A por otras del abecedario.)
DIN L(N) DIM T(N,M)	Dimensiona una variable de un índice. Dimensiona una variable de dos índices.
ENTER	Hace que el dato o instrucción tecleado se almacene en la memoria.
EXP X	Calcula la potencia e^x, siendo e = = 2.71828.
FNA (X)	Da el valor de la función de nombre FNA previamente definida, para el valor X.
FOR V=i TO t STEP s	Ejecuta un conjunto de instrucciones, llamado ***bucle,*** que empieza con la instrucción **FOR** y termina con una instrucción **NEXT.** (Variable: V; valor inicial de V, i; tope t; incremento, s.)
GOSUB n	Transfiere el control de la ejecución de una subrutina que empieza en una instrucción numerada con n.
GOTO n	Transfiere el control de la ejecución a la instrucción del programa numerado con n. Esta instrucción puede ser anterior o posterior a n.
IF (condición) THEN (instrucciones)	Si se cumple la ***condición*** entonces se ejecutan las instrucciones que siguen a **THEN;** y si no se cumple se ejecuta la instrucción escrita en la línea siguiente.
INPUT A INPUT A$	Permite introducir datos numéricos o alfamunéricos, los cuales se asignan a la variable A o A$, respectivamente.

Palabras BASIC	Función que realiza
INT X	Calcula la parte entera del número X.
LEN A$	Da el número de caracteres de la cadena almacenada en A$.
LET {variable}= ={expresión}	Calcula el valor de la expresión y lo asigna a la variable.
LIST LIST n	Hace aparecer en la pantalla: — todo el listado del programa almacenado en la memoria. — la instrucción n.
LN X	Calcula el logaritmo natural o neperiano de X.
NEW	Borra todo el contenido de la memoria.
NEXT V	Transfiere la ejecución del programa a la instrucción FOR del bucle siempre que el valor de V sea menor o igual que el tope del bucle.
PAUSE N	Detiene la ejecución del programa y retiene la imagen un tiempo que depende de N.
PEEK X	Da el contenido de la posición de la memoria de dirección X.
PI	Da el número 3.14159265 (número π).
POKE N, V	Almacena el valor de V en la posición de la memoria de dirección N (V es un entero comprendido entre 0 y 255).
PRINT	Escribe o imprime el valor de la variable, el resultado de operaciones aritméticas y texto entrecomillado.
RANDOMIZE y RND	Da un número aleatorio entre 0 y 1.

Palabras BASIC	Función que realiza
READ A, B, ..., X	Asigna a las variables A, B, ..., X los datos almacenados en una instrucción **DATA,** en forma secuencial.
REM	Permite añadir aclaraciones o comentarios en un programa sin afectar su ejecución.
RESTORE	Permite volver a leer desde el principio los datos contenidos en las instrucciones **DATA.**
RETURN	Última instrucción de una subrutina, cuya función es transferir el control de la ejecución siguiente a la GOSUB del programa principal.
RUN	Inicia la ejecución del programa. («Correr el programa».)
SGN X	Calcula el signo de X (+1, si X > 0; 0 si X = 0; −1 si X < 0).
SIN X	Calcula el seno del ángulo X, expresado en radianes.
SQR X	Calcula la raíz cuadrada de X.
STOP	Detiene la ejecución del programa justo en la línea en que se encuentra STOP.
STR$ N	Convierte el número almacenado en N en una cadena numérica.
TAB N	Mueve la posición del cursor a la columna N.
TAN X	Calcula la tangente de X, expresado en radianes.
TO	A$ (N TO M): extrae una subcadena de A$ desde la posición N a la M.
VAL N$	Convierte la cadena numérica almacenada en N$ en su valor numérico.

APÉNDICE II

INSTRUCCIONES PARA GRÁFICOS, COLORES Y MÚSICA

Instrucción	Significado	Formato	Función
ATTR	Atributo	ATTR (F, C)	Obtiene información sobre el color de la tinta y del papel, el brillo y el parpadeo de la posición de carácter de la fila F y columna C.
BEEP	Sonido	BEEP d, n	Genera una nota musical de duración d y de tono o altura correspondiente al número n.
BRIGHT	Brillo	BRIGHT b	Activa el brillo si b = 1 y lo desactiva si b = 0.
BORDER	Borde	BORDER n	Da al borde de la pantalla el color correspondiente al número n.
CIRCLE	Traza circunferencias	CIRCLE X, Y, R	Dibuja una circunferencia con centro en el punto (X, Y) y radio R.
DRAW	Traza líneas	DRAW H, V	Dibuja una línea recta desde el último punto que se imprimió o posicionó hasta el punto que se encuentra a H puntos en la horizontal y a V puntos en la vertical.
DRAW	Traza arcos	DRAW H, V, N	Si (A, B) son las coordenadas del último punto que se imprimió DRAW H, V, N, dibuja un arco que va desde (A, B) hasta (A + H, B + H) con un ángulo N; si éste es positivo genera el arco en sentido antihorario, y si es negativo, en sentido horario.
FLASH	Parpadeo, intermitente	FLASH b	Activa el parpadeo si b = = 1, y lo desactiva si b = 0.

INK	Tinta o primer plano	INK n	Da a la tinta el color correspondiente al número n.
INVERSE	Inverso, invertido	INVERSE b	Activa el invertido de tinta y papel si b = 1, y lo desactiva si b = 0.
OVER	Sobre	OVER b	Sirve para imprimir sobre un carácter sin borrarle (sirve para poner acentos).
PAPER	Papel o fondo	PAPER n	Da al papel el color correspondiente al número n.
PLOT	Imprime un punto	PLOT X, Y	Imprime un punto en el pixel de coordenadas referidas al vértice inferior izquierdo de la pantalla.
POINT	Punto	POINT (X, Y)	Comprueba si el pixel X, Y tiene color de tinta o de papel.

COLECCIÓN BASIC

Basic Programación

Gráficos, Colores y Música en el ZX Spectrum

MSX. Programación con Gráficos, Colores y Música

Programas resueltos en BASIC

Aplicaciones en MSX

distribuidor exclusivo cesma sa
C/ Aguacate, 25
28044 MADRID